Eberhard Karls Universität Tübingen





Fakultät für Informations- und Kognitionswissenschaften Wilhelm-Schickard-Institut

Workload Prediction für den WLM

Diplomarbeit

Verfasser:

Semmo Bakircioglu

31. Januar 2009

Betreuer:

Prof. Dr. Wilhelm G. Spruth Prof. Dr. Martin Bogdan Dipl.-Math. Peter Bäuerle (IBM)

Bakircioglu, Semmo:

Workload Prediction Diplomarbeit Informatik Eberhard Karls Universität Tübingen Bearbeitungszeitraum: 01.05.2008 - 31.01.2009

Ich erkläre hiermit, dass ich die vorliegende Arbeit selbständig verfasst und keine anderen als die angegebenen Quellen und Hilfsmittel verwendet habe.

Tübingen, den 25. Januar 2009

Semmo Bakircioglu

Kurzfassung

Diese Diplomarbeit führt das *Joint Research Project* "Workload Prediction auf Mainframes" der Universität Tübingen mit dem IBM Labor in Böblingen weiter. Im Rahmen dieses Projekts soll eine Machbarkeitsstudie zur möglichen Lastvorhersage mit Hilfe von künstlichen neuronalen Netzen erstellt werden. Als Ergebnis dieser Bemühungen entstand mit dem *WLP-Framework* ein erster Prototyp, der die gewünschte Funktionalität erstmals implementierte.

In der vorliegenden Arbeit wird die Weiterentwicklung dieses Prototypen thematisiert. Der besondere Fokus liegt dabei auf vier Punkten:

Als erstes wurde eine präzise und umfangreiche "Bedienungsanleitung" für das komplexe *WLP-Framework* erarbeitet, damit der Einstieg in das Thema nicht nur für die eigentliche Benutzung, sondern auch im Hinblick auf mögliche Weiterentwicklungen möglichst schnell und einfach erfolgen kann.

Im Anschluss wurde die ursprünglich verwendete *Apache Derby* Datenbank durch die von IBM entwickelte Datenbank *DB2 for z/OS* ersetzt. Eine zentrale Rolle spielt dabei die seit Version 9 eingeführte und so genannte *pureXML*-Technologie.

Durch diese Datenbankumstellung war es möglich den größten Nachteil des *WLP-Frameworks*, das *generische Tabellenformat*, zu beseitigen. Diese Aufgabe zog eine umfangreiche und aufwendige Refaktorisierung des Prototypen nach sich, in deren Zuge auch andere allgemeine Verbesserungen und Fehlerbehebungen am Quellcode vorgenommen wurden.

Nachdem mit der Datenbankumstellung ein erster Schritt weg vom Client unternommen wurde, folgte als letzter Schritt die Migration des für die Aufzeichnung zuständigen *RMFRecorders* und der für die Vorhersage zuständigen *WLPrediction* auf den Großrechner. Dazu wurden alle benötigten Dateien auf die *UNIX System Services* (USS) übertragen, und nach einigen notwendigen Anpassungen waren beide Anwendungen erfolgreich auf dem Großrechner ausführbar.

Somit ist mit Abschluss der vorliegenden Arbeit das komplette *WLP-Framework* explizit auch auf dem Großrechner lauffähig.

Danksagung

An dieser Stelle möchte ich all jenen danken, die eine Erstellung dieser Diplomarbeit erst ermöglicht haben. In erster Linie wären das Herr Prof. Dr. Spruth und Herr Prof. Dr. Bogdan als universitäre Betreuer sowie Herr Peter Bäuerle als Betreuer auf Seiten der IBM. Ohne sie wäre diese Diplomarbeit nie entstanden. Weiterhin danke ich dem kompletten Böblinger WLM-Team von Frau Jutta Brenner, die mir alle jederzeit und aufgeschlossen mit Rat und Hilfestellungen zur Seite standen. Vielen Dank Adrian, Hakan, Thomas, ... um nur ein paar wenige Namen zu nennen. Großer Dank gebührt auch Emil Wolf, der mir auf Nachfrage in Rekordzeit die benötigte Datenbank aufgesetzt und freundlicherweise auch die Administration übernommen hat.

Vor allem aber danke ich meiner Freundin Rine, die sich mit mir die Nächte um die Ohren geschlagen hat, um Korrektur zu lesen.

Inhaltsverzeichnis

Ab	Abbildungsverzeichnis v				
1.	Einl	itung	1		
	1.1.	Motivation	1		
	1.2.	Gliederung der Arbeit	2		
	1.3.	Ziele und Schwerpunkte dieser Arbeit			
	1.4.	Bisherige Arbeiten			
		1.4.1. Diplomarbeit von Clemens Gebhard	3		
		1.4.2. Diplomarbeit von Sarah Dorothea Kleeberg	4		
		1.4.3. Diplomarbeit von Michael Hagmann	5		
2.	Gru	dlagen: Die Komponenten des WLP-Frameworks	7		
	2.1.	Datenaufzeichnung: <i>RMFRecorder</i>	7		
	2.2.	Datenhaltung: <i>Apache Derby</i>	10		
	2.3.	Datenverarbeitung: WLPrediction	14		
3.	Weit	erentwicklung des WLP-Frameworks	19		
	3.1.	Erstellen einer Bedienungsanleitung	20		
	3.2.	Umstellen der Datenbank von <i>Apache Derby</i> auf <i>DB2 for z/OS</i>	23		
		3.2.1. DB2 <i>pureXML</i> -Technologie	23		
		3.2.2. Durchführung der Datenbankumstellung	26		
		3.2.3. Herstellen einer Datenbank-Verbindung	27		
		3.2.3.1. IBM Data Studio Developer	28		
		3.2.3.2. <i>SQuirreL SQL</i>	30		
	3.3.	Elimination des <i>generischen Tabellenformats</i>	32		
		3.3.1. Möglichkeiten der XML-Verarbeitung in Java	32		
		3.3.2. XML-Verarbeitung mit <i>JDOM</i>	35		
		3.3.3. Refaktorisierung des <i>RMFRecorders</i>			

		3.3.4. Refaktorisierung der <i>WLPrediction</i>	40			
	3.4.	Migration des WLP-Frameworks auf den Großrechner				
4.	Erge	ebnisse	49			
	4.1.	Bedienungsanleitung	49			
	4.2.	Datenbankumstellung auf <i>DB2</i> for <i>z</i> /OS	50			
	4.3.	Elimination des generischen Tabellenformats	50			
	4.4.	Migration auf den Großrechner	52			
5.	Zusa	ammenfassung und Ausblick	55			
Α.	Bedi	enungsanleitung für das WLP-Framework	59			
В.	Inhalt der beigefügten DVD 11					
Lit	teraturverzeichnis 11					

Abbildungsverzeichnis

2.1.	Schematische Übersicht über die Komponenten des WLP-Frameworks	
	(Abbildung aus [Hag07])	8
2.2.	Abbildung von RMF Metrik-Elementen auf CIM-Elemente (Abbil-	
	dung aus [Hag07])	9
2.3.	CIMReader Konfigurationsdatei (cimmetrics.xml)	11
2.4.	Schematische Übersicht des <i>RMFRecorder</i> (Abbildung aus [Hag07]) .	11
2.5.	Relationales Tabellenformat (Abbildung aus [Hag07])	13
2.6.	Generisches Tabellenformat (Abbildung aus [Hag07])	13
2.7.	Konvertierung vom generischen in das von der WLPrediction-	
	Anwendung benötigte relationale Tabellenformat (Abbildung aus	
	[Hag07])	14
2.8.	ABLE-Editor mit erstelltem WlpAgent	16
2.9.	Graphische Darstellung der Vorhersagewerte mit Hilfe von Jetty und	
	JCharts	17
3.1.	Ablaufdiagramm für die Inbetriebnahme des <i>RMFRecorder</i>	21
3.1. 3.2.	Ablaufdiagramm für die Inbetriebnahme des <i>RMFRecorder</i> Ablaufdiagramm für die Inbetriebnahme der <i>WLPrediction</i>	21 22
 3.1. 3.2. 3.3. 	Ablaufdiagramm für die Inbetriebnahme des <i>RMFRecorder</i> Ablaufdiagramm für die Inbetriebnahme der <i>WLPrediction</i> Hybride Datenbank DB2 9: Kombinierte Speicherung von XML-	21 22
3.1.3.2.3.3.	Ablaufdiagramm für die Inbetriebnahme des <i>RMFRecorder</i> Ablaufdiagramm für die Inbetriebnahme der <i>WLPrediction</i> Hybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])	21 22 25
3.1.3.2.3.3.3.4.	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio Startbildschirm	21 22 25 29
 3.1. 3.2. 3.3. 3.4. 3.5. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL Startbildschirm	21 22 25 29 31
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])	21 22 25 29 31 34
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 3.7. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6	21 22 25 29 31 34
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 3.7. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6(Abbildung aus [Wor08b])	21 22 25 29 31 34
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 3.7. 3.8. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6(Abbildung aus [Wor08b])Das alte generische Tabellenformat (Screenshot aus SQuirreL)	21 22 25 29 31 34 34
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 3.7. 3.8. 3.9. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6(Abbildung aus [Wor08b])Das alte generische Tabellenformat (Screenshot aus SQuirreL)Das neue XML-Tabellenformat (Screenshot aus SQuirreL)	21 22 25 29 31 34 34 38 39
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 3.7. 3.8. 3.9. 3.10. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6(Abbildung aus [Wor08b])Das alte generische Tabellenformat (Screenshot aus SQuirreL)Das neue XML-Tabellenformat (Screenshot aus SQuirreL)XML-Dokument mit aufgezeichneten Metriken (erstellt mit Hilfe	 21 22 25 29 31 34 34 38 39
 3.1. 3.2. 3.3. 3.4. 3.5. 3.6. 3.7. 3.8. 3.9. 3.10. 	Ablaufdiagramm für die Inbetriebnahme des RMFRecorderAblaufdiagramm für die Inbetriebnahme der WLPredictionHybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])Data Studio StartbildschirmSQuirreL StartbildschirmXHTML-Tabelle (Abbildung aus [Wor08b])Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6(Abbildung aus [Wor08b])Das alte generische Tabellenformat (Screenshot aus SQuirreL)Das neue XML-Tabellenformat (Screenshot aus SQuirreL)XML-Dokument mit aufgezeichneten Metriken (erstellt mit Hilfevon JDOM)	21 22 25 29 31 34 34 38 39 40

3.11.	Übersicht über den Inhalt des dist-Verzeichnisses	 •	•••	•	 		46
B.1.	Inhalt der beigefügten DVD	 •		•	 	1	12

1. Einleitung

Um den Umfang dieser Arbeit zu beschränken wurde bewusst darauf verzichtet die theoretischen Grundlagen zu sehr zu vertiefen, da dies in den vorhergehenden Diplomarbeiten dieses Projekts ausführlich erledigt wurde.

Falls Interesse an tiefer gehenden Informationen zum Thema künstliche neuronale Netze besteht, wird auf die Arbeit von Kleeberg [Kle06] verwiesen. Für Informationen zur Datengewinnung und -verarbeitung wird Gebhard [Geb06] empfohlen. Alternativ kann auch die Arbeit von Hagmann [Hag07] als komprimierte Zusammenfassung der vorher genannten angesehen werden.

In allen drei Arbeiten finden sich weitere Quellenangaben für zusätzliche Informationen.

1.1. Motivation

Zentrales Thema dieser Arbeit ist die Lastvorhersage auf IBM Großrechnern. Die Verwaltung von Betriebsmitteln solcher Systeme erfolgt über den *Workload Manager* (WLM), der integraler Bestandteil des Betriebssystems z/OS ist. Bisher wird diese Verwaltung dynamisch nach benutzerdefinierten Zielvorgaben durchgeführt. Die Basis dafür bilden aktuell gemessene Systemdaten sowie Daten aus der Vergangenheit. Der WLM ist eine zentrale Komponente in der alltäglichen Arbeit eines Großrechner, da ein priorisierter Zugang zu Betriebsmitteln zwingend erforderlich ist. Diese Priorisierung ist notwendig, weil Großrechner darauf ausgelegt sind eine Vielzahl an unterschiedlichen Anwendungen von verschiedenen Benutzern zeitgleich und ohne spürbare Geschwindigkeitseinbußen auszuführen.

Durch seine zielorientierte Arbeitsweise sind dem WLM aber Grenzen gesetzt. Es kann immer nur zeitverzögert auf Ressourcen-Engpässe reagiert werden, da sie erst nach ihrem tatsächlichen Eintreten bemerkt werden können. Mit dieser Arbeitsweise sind Engpässe unvermeidbar. Um diesen Schwachpunkt mittelfristig zu beseitigen und Engpässe möglichst im voraus erkennen zu können, wurde dieses Projekt mit dem Ziel einer Lastvorhersage mit Hilfe von künstlichen neuronalen Netzen gestartet.

Durch die Einführung des *Capacity Provisionings* (CP) mit z/OS 1.9 gewann dieser Ansatz an zusätzlichem Gewicht. Der CP-Manager hat Zugriff auf die Systemdaten der *Resource Measurement Facility* (RMF). Sobald klar wird, dass eine Anwendung ihr festgelegtes WLM-Ziel verfehlt weil ein Ressourcen-Engpass besteht, kann der CP-Manager entweder angemessene Maßnahmen vorschlagen oder automatisch zusätzliche Kapazitäten aktivieren [IBM08d].

Ziel dieses Projekts ist eine Anwendung zur Lastvorhersage mit Hilfe von künstlichen neuronalen Netzen, die in die bestehende WLM- und CP-Umgebung integriert werden soll. Mit Hilfe der vorhergesagten Werte sollen mögliche Engpässe bereits im Voraus erkannt und an das CP weitergegeben werden, damit es je nach Bedarf vorhandene stillgelegte Betriebsmittel zur richtigen Zeit aktivieren kann.

1.2. Gliederung der Arbeit

Kapitel 1 dient als Motivation und Einstieg in das Thema. Zusätzlich werden die Ziele und Schwerpunkte dieser Arbeit erläutert und es wird auf die bisherigen Diplomarbeiten jeweils kurz eingegangen um einen groben chronologischen Überblick über das ganze Projekt zu bekommen. Kapitel 2 stellt die Arbeit von Michael Hagmann [Hag07] detaillierter vor, welche mit dem *WLP-Framework* einen ersten Software-Prototypen für das *Joint Research Project* umfasst. Es bildet die Grundlagen für die anschließenden Weiterentwicklungen in Kapitel 3, dem eigentlichen Kern dieser Arbeit. Hier wird dann, beginnend mit dem Erstellen einer Bedienungsanleitung und einer Datenbankumstellung, die Migration auf den Großrechner durchgeführt. Zudem erfolgt vor der Migration eine Refaktorisierung, die strukturelle Optimierungen zur Folge hat. Anschließend werden in Kapitel 4 die erzielten Ergebnisse rekapituliert und diskutiert, bevor in Kapitel 5 eine kurze Zusammenfassung mit Ausblick auf zukünftige Forschungsarbeiten im Rahmen dieses Projekts die Arbeit abschließt.

1.3. Ziele und Schwerpunkte dieser Arbeit

Ziel dieser Diplomarbeit ist die Weiterentwicklung der Machbarkeitsstudie von Hagmann [Hag07]. Dieser hat einen ersten Prototypen zur Lastvorhersage mit künstlichen neuronalen Netzen für IBMs Großrechner *System Z* implementiert. Im Verlauf der vorliegenden Arbeit liegt der Fokus vor allem auf vier Punkten:

- Erstellen einer präzisen und umfassenden Bedienungsanleitung zur Inbetriebnahme und zum allgemeinen Verständnis des Prototypen
- Ersetzen der aktuell benutzten *Apache Derby* Datenbank durch IBM *DB2 for z/OS* 9.1
- Elimination des *generische Tabellenformats* mit Hilfe der neuen DB2 *pureXML*-Technologie und eine dadurch bedingte umfangreiche Refaktorisierung großer Teile des *WLP-Frameworks*
- Migration des WLP-Frameworks vom Windows-Client auf den Großrechner

1.4. Bisherige Arbeiten

Dies ist die vierte Diplomarbeit im Rahmen des *Joint Research Projects* "Workload Prediction auf Mainframes". Nach anfangs theoretischen Arbeiten über "Datengewinnung und Parameterbestimmung zur Lastvorhersage in z/OS" von Gebhard [Geb06] und zum Thema "Neuronale Netze und Maschinelles Lernen zur Lastvorhersage in z/OS" von Kleeberg [Kle06] erfolgte im Anschluss der Versuch einer ersten lauffähigen Umsetzung der Ergebnisse in der Arbeit "Architektur und Integration der Workload-Vorhersage auf Basis neuronaler Netze in z/OS" von Hagmann [Hag07]. Auf die Themen dieser Diplomarbeiten wird im Folgenden kurz eingegangen.

1.4.1. Diplomarbeit von Clemens Gebhard

In der Arbeit von Clemens Gebhard [Geb06] wurde in erster Linie die Beschaffung von Systemdaten untersucht: Zum einen die Gewinnung von realen Daten und zum anderen die Generierung von künstlichen Daten. Ziel war es möglichst realitätsnahe Daten für die Durchführung einer Lastvorhersage mit Hilfe von künstlichen neuronalen Netzen zu erhalten.

Ein weiterer wichtiger Aspekt war die Aufzeichnung und Weiterverarbeitung der Daten. Hierzu wurde erstens die Abfrage über die RMF und zweitens über die *System Management Facility* (SMF) betrachtet. Abschließend wurden verschiedene Techniken der Vorverarbeitung für die extrahierten Daten erörtert, da die Daten in der Regel Ausreißer und fehlende Werte beinhalten, welche entsprechend skaliert, gefiltert oder anderweitig bearbeitet werden müssen.

Zudem wurden erste Ansätze zur Implementierung der erarbeiteten Ergebnisse präsentiert, mit denen die Daten graphisch aufbereitet und analysiert werden konnten.

Die Ergebnisse dieser Arbeit bilden den theoretischen Hintergrund des implementierten *WLP-Frameworks* zum Thema Datengewinnung und -verarbeitung.

1.4.2. Diplomarbeit von Sarah Dorothea Kleeberg

Die Arbeit von Sarah Kleeberg befasst sich schwerpunktmäßig damit, die Leistungsfähigkeit von unterschiedlichen Vorhersagemodellen zu betrachten, um in der Folge abwägen zu können welche Modelle für welchen Vorhersagezeitraum geeignet wären. Im Detail wurden die künstlichen neuronalen Netztypen *Feed-Forward* und *FlexNet* sowie der maschinelle Lernalgorithmus *Support Vector Regression* bezüglich ihrer Leistung durch mehrere Testläufe unter verschiedenen Voraussetzungen verglichen. Die Modell-Bewertung lief dabei über die Betrachtung von verschiedenen Fehlermaßen. Darunter auch der im Prototyp verwendete *Mean Squared Error with Regularization* (MSEREG).

Als Ergebnis dieser Testläufe wurde festgehalten, dass für längerfristige Vorhersagen *Feed-Forward* Netze und *FlexNet* die besten Ergebnisse liefern. *Support Vector Regression* war nur bei kürzeren Zeiträumen mit ungeglätteten Datensätzen die überlegenere Methode.

Als Schlussfolgerung aus dem oben genannten Verhalten und der detaillierten Betrachtung der restlichen Ergebnisse wurde die Erkenntnis gewonnen, dass für unterschiedliche Problemstellungen auch unterschiedliche Vorhersagemodelle aus einem "Pool" zur Auswahl stehen sollten. Die Vorhersage-Anwendung, die die Komponente mit den künstlichen neuronalen Netzen des *WLP-Frameworks* implementiert wurde nach den Ergebnissen dieser theoretischen Arbeit modelliert.

1.4.3. Diplomarbeit von Michael Hagmann

Nachdem in den beiden vorherigen Arbeiten die theoretischen Grundlagen geschaffen wurden, war der Schwerpunkt der Arbeit von Hagmann [Hag07] das Herausarbeiten einer Architektur, die diese gewonnen Erkenntnisse zu einer lauffähigen Anwendung kombiniert und möglichst ohne Nachteile für das Gesamtsystem implementiert.

Der erste Schritt dazu war die Datengewinnung und -speicherung. Es musste ein Datenzugriff bei laufender Anwendung ermöglicht werden und für die weitere Bearbeitung musste ein Weg gefunden werden die Daten möglichst vorteilhaft zwischenzuspeichern. Die einzige plausible und zeitgemäße Lösung für diese Aufgabe war eine relationale Datenbank.

Ein weiterer entscheidender Aspekt betraf die automatische Merkmalsselektion. Es sollte eine möglichst große Auswahl an Metriken (Merkmalen) zur Verfügung stehen, um je nach Vorhersagemodell die am besten geeigneten Merkmale auch selektieren zu können. Um je nach Bedarf trotzdem eine Vorauswahl der Metriken zu treffen, wurde eine Konfigurationsdatei eingeführt, in der mit *Include/Exclude-*Anweisungen Metriken ein-/ausgeschlossen werden können.

Um das *WLP-Framework* möglichst flexibel zu gestalten, musste eine Möglichkeit gefunden werden mehrere Vorhersagemodelle erstens unabhängig voneinander und zweitens parallel zu betreiben und zu verwalten. Damit war gewährleistet, dass man während des Trainierens eines neu angelegten Netzes gleichzeitig eine Vorhersage mit einem schon trainierten Netz durchführen kann. In der Folge wurden der *RMFRecorder* zur Datenaufzeichnung und die *WLPrediction* zur Vorhersageberechnung als komplett eigenständige und unabhängige Anwendungen implementiert.

Als letzten Schritt wurde zudem noch mit Hilfe eines Webservers eine einfach gehaltene Schnittstelle implementiert, die die Vorhersagewerte in graphischer Form anschaulich präsentiert, und die zukünftig dazu beitragen könnte die Anwendung zu steuern und zu konfigurieren.

2. Grundlagen: Die Komponenten des *WLP-Frameworks*

Als Grundlage für die vorliegende Arbeit dient das in [Hag07] von Hagmann erstellte *WLP-Framework*. Bevor auf die eigentlichen Aufgaben dieser Arbeit eingegangen wird, soll deswegen zuerst dieses als Ausgangspunkt dienende Framework im Detail vorgestellt werden.

Das WLP-Framework basiert auf einer verteilten Architektur, die in Abbildung 2.1 schematisch dargestellt wird. Sie kann grob in drei Hauptkomponenten aufgeteilt werden: Den *RMFRecorder* für die Datenaufzeichnung, die Datenbank für die Datenhaltung und die *WLPrediction* für die eigentliche Vorhersage. Alle drei Komponenten werden auf einem Windows-Client ausgeführt.

Die Architektur wurde auf möglichst viel Flexibilität ausgelegt. Es ist möglich, die Datenaufzeichnung und die Vorhersage getrennt voneinander durchzuführen, da sie beide als unabhängige und eigenständige Anwendungen konzipiert wurden. Auch ein Parallelbetrieb ist jederzeit möglich. Zudem ist das in Java implementierte *WLP-Framework* theoretisch plattformunabhängig, da es aber auf der Windows-Plattform erstellt wurde, ist es momentan auch nur darauf getestet worden. Im Folgenden werden die Komponenten jeweils kurz betrachtet.

2.1. Datenaufzeichnung: RMFRecorder

Wie bereits erwähnt, ist der *RMFRecorder* von der Vorhersage-Anwendung unabhängig, da er in einem eigenen Prozess läuft. Der Vorteil dabei besteht in der Möglichkeit erst einmal nur Daten aufzuzeichnen und zu sammeln, während die eigentliche Vorhersage u. U. erst viel später erfolgt. Darüber hinaus kann eine Vorhersage auch parallel zu einer laufenden Datenaufzeichnung durchgeführt werden.



Abbildung 2.1.: Schematische Übersicht über die Komponenten des *WLP-Frameworks* (Abbildung aus [Hag07])

Als Datenquelle für den *RMFRecorder* dient RMF, da sich herausgestellt hat, dass sämtliche für die Vorhersage benötigten Daten im RMF-Datenbestand enthalten sind [Hag07]. Das Aufzeichnen der Daten erfolgt über so genannte *Reader*-Module. Dabei ist es möglich mehrere verschiedene solcher *Reader* zu implementieren und je nach Wunsch den jeweiligen zum Aufzeichnen zu nutzen. Diese Auswahl muss über die Haupt-Konfigurationsdatei WLPDefaults.properties vorgenommen werden.

Aktuell ist aber nur der *CIMReader* implementiert, mit dem die Aufzeichnung der Metriken über das von der *Distributed Management Task Force* standardisierte *Common Information Model* (CIM) erfolgt. Bei CIM handelt es sich um einen offenen Standard, der Definitionen für das Management von IT-Systemen, Netzwerken, Anwendungen und Services bereitstellt. Mit diesen standardisierten Definitionen wird der Austausch von einheitlichen Informationen über Management-Schnittstellen zwischen verschiedenen Systemen anbieter- und plattformunabhängig ermöglicht [Dis08a]. CIM ist weitverbreitet und wird u. a. auch von CP verwendet, welches in Zukunft idealerweise mit dem *WLP-Framework* kommunizieren soll. Auch deswegen fiel in [Hag07] die Wahl auf den CIM-Standard.

Um die RMF-Metriken in CIM abzubilden, wurden die CIM-Standardtypen



Abbildung 2.2.: Abbildung von RMF Metrik-Elementen auf CIM-Elemente (Abbildung aus [Hag07])

$\texttt{CIM}_\texttt{BaseMetricDefinition} \ und \ \texttt{CIM}_\texttt{BaseMetricValue} \ zu$

IBMzOS_BaseMetricDefinition und IBMzOS_BaseMetricValue erweitert. Die Beziehung zwischen Metrik-Definition und Metrik-Wert wird dabei über so genannte Associations definiert. Zusätzlich sollten die Definition-Werte-Paare auch mit einer Instanz von CIM_ManagedElement verknüpft sein. Dieser Zusammenhang wird in Abbildung 2.2 nochmals veranschaulicht.

Die Bereitstellung von Daten mit Hilfe des CIM-Datenmodells erfolgt über eine *Client/Server*-Architektur, in welcher der Server Daten bereitstellt auf die der Client jederzeit durch Anfragen zugreifen kann.

Auf dem Server kommt *OpenPegasus* zum Einsatz. Es handelt sich dabei um ein Open Source Projekt der *Open Group* das sich aus CIM-Server, *Object Broker* (CIMOM) und CIM-Provider für die wichtigsten Klassen zusammen setzt. Es ist in C++ implementiert um möglichst schnell und effizient zu sein. Zusätzlich ist es darauf ausgelegt möglichst modular und portabel zu sein [Ope08]. *OpenPegasus* ist in den neueren z/OS-Releases bereits integriert.

Auf dem Client findet der aus dem Open Source Projekt Standards Based Linux Instrumentation for Manageability (SBLIM) stammende SBLIM CIM Client for Java Verwendung. Auch der Client wurde auf Geschwindigkeit und Effizienz optimiert, bietet zusätzlich aber auch Vorteile beim Debuggen durch Bereitstellen von *Logfiles* [Sta08].

Als Übertragungsprotokoll für den Austausch zwischen Client und Server wird das ebenfalls von der DMTF standardisierte CIM-XML verwendet. Es kapselt die Daten plattformunabhängig in XML und benutzt zur Übertragung eine TCP/HTTP-Verbindung [Dis08b].

Die über CIM abgefragten Metriken speichert der *RMFRecorder* zur weiteren Verarbeitung durch die Vorhersage-Anwendung in einer relationalen Datenbank ab. Da selten alle vorhandenen Metriken benötigt werden und eine komplette Speicherung über Wochen oder Monate hinweg zu viel Speicherplatz in Anspruch nehmen könnte, besteht die Möglichkeit einer Merkmalsselektion über die Konfigurationsdatei *cimmetrics.xml*. Diese spezielle Auswahl der gewünschten Metriken erfolgt dabei mit Hilfe von *Include/Exclude-*Anweisungen, die bestimmte Komponenten modular von vornherein ein- oder ausschließen. Ein mögliches Beispiel für eine solche Konfigurationsdatei findet sich in Abbildung 2.3.

Weitere Konfigurationsmöglichkeiten für den *RMFRecorder* lassen sich in der Haupt-Konfigurationsdatei WLPDefaults.properties vornehmen. Neben der Möglichkeit die Stützstellen der Zeitreihen durch *Downsampling* zu reduzieren und dadurch weiter Speicherplatz zu sparen wird dort u. a. die Datenbank näher spezifiziert, festgelegt in welchem Zeit-Intervall Aufzeichnungen erfolgen sollen und das oben erwähnte Modul des *Readers* ausgewählt.

Der schematische Aufbau des *RMFRecorders* wird in Abbildung 2.4 dargestellt. Hier wird neben dem aktuell implementierten *CIMReader*-Modul auch das optionale *DDSReader*-Modul abgebildet, welches die RMF-Metriken mit Hilfe des *Dedicated Data Server* (DDS) aufzeichnet.

2.2. Datenhaltung: Apache Derby

Ein wichtiger Aspekt bei der Vorhersage mit künstlichen neuronalen Netzen ist die Datenhaltung. Je mehr Daten für das Training des Netzes zur Verfügung stehen, desto höher die Chancen auf gute Vorhersage-Ergebnisse. Da diese Daten optimalerweise auch über einen längeren Zeitraum verfügbar sein müssen, wurden sie in einer relationalen Datenbank gespeichert.

🗷 C:\Documents and Settings\Administrator\My Documents\Diplomarbeit\workspace\WLPrediction\cimmetrics.xml - N 🖃 🗖 🔀								
File	File Edit Search View Format Language Settings Macro Run TextFX Plugins Window ? X							
	6	86	🕞 🔓 🖨 🐇 🛍 🛅 🤿 🗲 🟙 🋬 🍳 👒 🖳 🔤 🎫 🎙 🗐 🗐 🔍 🗩 📾 🗟 H 🛁 🗔 🤝					
	cimmet	trics.xml						
	1	x</th <th>ml version="1.0" encoding="UTF-8"<mark>?></mark></th>	ml version="1.0" encoding="UTF-8" <mark>?></mark>					
	2	<mm< th=""><th>::MonitoredMetrics xmlns:mm="http://www.ibm.com/rmf/gpm/cim/metrics"</th></mm<>	::MonitoredMetrics xmlns:mm="http://www.ibm.com/rmf/gpm/cim/metrics"					
	3		<pre>xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"</pre>					
	4	Ę	<pre>xsi:schemaLocation="http://www.ibm.com/rmf/gpm/cim/metrics cimmetrics.xsd"></pre>					
	5	¢.	<ressource></ressource>					
	6		<caption>CEC</caption>					
	7		<cim_classname>IBMz_CEC</cim_classname>					
	8	¢.	<excludes></excludes>					
	9	¢.	<metric></metric>					
	10		<caption>Number of defined CPs</caption>					
	11		<name>NumberOfDefinedCPs</name>					
	12	-						
	13	¢.	<metric></metric>					
	17		<metric></metric>					
	21	¢.	<metric></metric>					
	25		<metric></metric>					
	29		<metric></metric>					
	35		<metric></metric>					
	41	¢.	<metric></metric>					
	45		<metric></metric>					
	49							
	50	-						
	51	÷.	<ressource></ressource>					
	55	¢.	<ressource></ressource>					
	65	÷.	<ressource></ressource>					
	69	L <th>m:MonitoredMetrics></th>	m:MonitoredMetrics>					
, eXter	sible M	1arkup La	nguage file nb char : 1947 Ln : 69 Col : 23 Sel : 0 Dos\Windows ANSI INS					

Abbildung 2.3.: CIMReader Konfigurationsdatei (cimmetrics.xml)



Abbildung 2.4.: Schematische Übersicht des RMFRecorder (Abbildung aus [Hag07])

Für diese Zwecke fiel die Wahl auf die *Apache Derby* Datenbank, der Open Source Variante der mittlerweile eingestellten *IBM Cloudscape* Datenbank. Genauso wie der *RMFRecorder* läuft auch diese auf dem Client. Vorteile von *Apache Derby* sind die leichtgewichtige Implementation, die freie Verfügbarkeit und der eingebettete Treiber für *Java Database Connectivity* (JDBC). Da sie zusätzlich in Java implementiert ist und durch ihre Architektur – alle Daten werden in einer Verzeichnisstruktur und in *dat*-Dateien gespeichert – betriebssystemunabhängig ist, stellt sie eine hoch portable und flexible Datenbank dar, die relativ schnell in jede beliebige Java-Anwendung eingebettet werden kann. Schließlich besitzt *Apache Derby* auch noch einen Netzwerk-JDBC-Modus der im vorliegenden Fall unentbehrlich ist, da die benötigten Metriken über eine *Client/Server*-Architektur direkt vom Großrechner über das Netzwerk auf den Client zurückgeschrieben werden [Apa08].

Die erste große Hürde bei der Implementierung stellte das Datenbankdesign dar. Nach dem relationalen Modell hätte jede aufgezeichnete Metrik jeweils eine eigene Spalte bekommen. Durch die festgelegte Vorgabe nach maximaler Flexibilität besaß dieser Ansatz aber einen bedeutenden Nachteil. In der Folge hätte dies dazu führen können, dass bei jeder Aufzeichnung eine unterschiedliche Anzahl von Spalten verwendet worden wäre. Zudem könnte die selbe Metrik in verschiedenen Aufzeichnungen jeweils in unterschiedlichen Spalten gespeichert worden sein.

Unter den gegebenen Voraussetzungen war dieser Ansatz nicht realisierbar, da sämtliche in der *WLPrediction*-Anwendung eingebetteten SQL-Abfragen je nach vorliegendem Fall angepasst werden müssten. Um dieses Problem zu lösen, hat sich Hagmann in [Hag07] eines "Tricks" bedient und das so genannte *generische Tabellenformat* eingeführt: Die Metrik-Bezeichnungen werden als Teil des Primärschlüssels verwendet und alle Metriken kommen in die selbe Spalte. Somit ist die Anzahl der Spalten fix, die Werte dieser Metrik-Spalte sind aber variabel und jederzeit problemlos erweiterbar. Abbildungen 2.5 und 2.6 erläutern diese Idee anschaulich. Dieses *generische Tabellenformat* ermöglicht es weiterhin den Plan einer flexiblen Architektur mit automatischer Merkmalsselektion zu realisieren. Damit ist es theoretisch möglich, eine größere Anzahl an Metriken zu speichern und für den jeweils vorliegenden Fall das optimale Vorhersagemodell mit den dafür benötigten Metriken auszuwählen.

Die Metriken werden dazu in der Tabelle metrics gespeichert, die wie in [Hag07] beschrieben mit dem folgenden SQL-Befehl erstellt wurde:

```
CREATE TABLE metrics
(
   TimeStamp TIMESTAMP NOT NULL,
   InstanceId VARCHAR (50) NOT NULL,
   MetricId VARCHAR (20) NOT NULL,
   MetricValue VARCHAR (20) NOT NULL,
   DataType SMALLINT NOT NULL,
   PRIMARY KEY (TimeStamp, InstanceId)
);
CREATE INDEX met_id ON metrics (MetricId);
```

Der Schlüssel setzt sich aus dem Zeitstempel und der Instanz-ID zusammen. Da aber auch eine Suche nach der Metrik in manchem Fall sinnvoll sein könnte wurde darauf zusätzlich ein Index erstellt um potentielle Abfragen darauf zu beschleunigen.

Abschließend betrachtet, stellte sich das *generische Tabellenformat* als sehr großer Nachteil bei der Implementierung der *WLPrediction*-Anwendung heraus. Hauptproblem war die Reihenfolge der in der Datenbank gespeicherten Metriken. Da der Primärschlüssel aus einer Kombination der Spalten *TimeStamp* und *InstanceId* bestand, konnte diese Reihenfolge nicht garantiert werden. Bei künstlichen neuronalen Netzen ist dieser Punkt aber zwingende Voraussetzung für eine korrekte Funktionsweise, da die Metriken immer in der selben Reihenfolge auf den Eingangsneuronen anliegen müssen. Diese Tatsache führte bei der Implementierung zu großen Problemen und hatte die Entwicklung eines Hilfskonstrukt zur Folge, welches die Komplexität des *WLP-Frameworks* deutlich erhöhte.

Ein Teil dieser Hilfkonstruktion war z. B. für die Konvertierung vom generischen in

D	Spalte 1	Spalte 2	Spalte 3
1	Wert (1,1)	Wert (1,2)	Wert (1,3)
2	Wert (2,1)	Wert (2,2)	Wert (2,3)
3	Wert (3,1)	Wert (3,2)	Wert (3,3)

Abbildung 2.5.: Relationales Tabellenformat (Abbildung aus [Hag07])

ID	Spalte	Wert
1	Spalte 1	Wert (1,1)
1	Spalte 2	Wert (1.2)
1	Spalte 3	Wert (1,3)
2	Spalte 1	Wert (2,1)
2	Spalte 2	Wert (2,2)
2	Spalte 3	Wert (2,3)
3	Spalte 1	Wert (3,1)
3	Spalte 2	Wert (3,2)
3	Spalte 3	Wert (3,3)

Abbildung 2.6.: *Generisches Tabellenformat* (Abbildung aus [Hag07])



Abbildung 2.7.: Konvertierung vom *generischen* in das von der *WLPrediction*-Anwendung benötigte relationale Tabellenformat (Abbildung aus [Hag07])

das relationale Tabellenformat zuständig. Abbildung 2.7 stellt diesen Teil schematisch dar.

Auf die Beseitigung dieses gravierenden Nachteils wird in Kapitel 3 dieser Arbeit detailliert eingegangen.

2.3. Datenverarbeitung: WLPrediction

Die dritte und letzte Komponente des *WLP-Frameworks* bildet die *WLPrediction*-Anwendung. Aufgabe dieser Komponente ist die Generierung von Vorhersagen mit den vorher durch den *RMFRecorder* aufgezeichneten Metriken. Diese Vorhersagen werden mit Hilfe von künstlichen neuronalen Netzen berechnet. Wie die beiden Komponenten davor, läuft auch *WLPrediction* auf dem Client-Rechner.

Kernstück der Anwendung ist das *Agent Building and Learning Environment* (ABLE), welches für die Umsetzung des künstlichen neuronalen Netzwerks zuständig ist. Dabei handelt es sich um ein Projekt im Rahmen des *IBM alphaWorks* Programm das eine Java-Entwicklungsumgebung und Komponentenbibliothek für die Entwicklung von Anwendungen und so genannten "Agenten" basierend auf Elementen des maschinellen Lernens und der Künstlichen Intelligenz bereitstellt [IBM08a]. Diese Funktionen wurden in ABLE-Beans realisiert und setzen sich laut [IBM08a] im Wesentlichen wie folgt zusammen:

- Data Beans
- Learning Beans
- Rule Beans
- ABLE Agents

Durch das Beans-Konzept ist es möglich mehrere ABLE-Beans zu Agenten zusammenzufassen oder gar mehrere Agenten zu einem neuen Agenten zu verknüpfen. Zusätzlich zu den schon vorhandenen ABLE-Beans ist es möglich eigene Beans mit speziell benötigten Funktionen zu implementieren und in das *WLP-Framework* einzubinden. Dies ist auch in der *WLPrediction* des vorliegenden Prototypen so geschehen. Der implementierte *WlpAgent* besteht aus den folgenden Beans:

- TrainingImport (Trainingsdaten)
- TestImport (Testdaten)
- OnlineImport (Onlinedaten)
- InFilter (Skalierung der Eingangsdaten)
- MovingAverageFilter (Glätten der Daten mit gleitendem Mittelwert)
- TimeSeriesFilter (Mapping der Metriken auf die Eingangsneuronen)
- BackPropagation (Lernalgorithmus für das Feed-Forward Netzwerk)
- OutFilter (Rückskalierung der Ausgangsdaten)

Sämtliche Beans wurden entweder neu implementiert oder sind aus den schon vorhandenen ABLE-Beans durch Erweiterung und Anpassung entstanden. Dabei kamen die Ergebnisse aus den theoretischen Arbeiten [Kle06] und [Geb06] zum Einsatz.

WLPrediction benötigt für die Vorhersagen einen trainierten *WlpAgent*, den es aus einem vorher festgelegten AgentPool bezieht. Das Erstellen, Trainieren und Speichern des jeweiligen Agenten ist Aufgabe des Benutzers und muss vor der Ausführung von *WLPrediction* erfolgen. Diese Aufgabe wird mit dem *ABLE-Editor* erledigt, der Teil des *ABLE-Frameworks* ist und im *Eclipse-Workspace* als *ablegui*-Projekt zu finden ist. Ein im *ABLE-Editor* frisch angelegter *WlpAgent* wird in Abbildung 2.8 dargestellt:



Abbildung 2.8.: ABLE-Editor mit erstelltem WlpAgent

Bei der Erstellung des *WlpAgents* werden neben den Angaben zur Datenbank und den Tabellen unter anderem auch die gewünschten Metriken, Intervalle und Vorhersageschritte angegeben. Mehr Details zur Erstellung und zum Trainieren von Agenten mit Hilfe des *ABLE-Editors* folgen in Kapitel 3.1 und in der Bedienungsanleitung in Anhang A.

Die WLPrediction-Anwendung speichert als Ausgabe drei Werte in die Datenbank, genauer in die Tabelle prediction. Neben dem tatsächlichen Wert der einem bestimmten Zeitstempel zugeordnet ist, wird für jeden Zeitstempel auch noch der mit dem gleitenden Mittelwert geglättete Wert zurückgeschrieben. Dieser geglättete Wert dient dann auch als Zielwert für den dritten und letzten Wert. Dabei handelt es sich um den Vorhersagewert, der um die im *WlpAgent* festgelegten Zeitschritte verschoben wird.

Durch das Speichern aller drei Werte ist es möglich die Vorhersagen zu validieren und nach dem Eintreten der tatsächlichen Werte je nach Qualität der Vorhersagen bestimmte Aktionen manuell oder automatisiert durchzuführen. Dies ist bisher aber nur angedacht und noch nicht implementiert worden. Denkbar wären z. B. folgende Aktionen: Ein Nachtrainieren des Agenten, das Ändern der verwendeten



Abbildung 2.9.: Graphische Darstellung der Vorhersagewerte mit Hilfe von *Jetty* und *JCharts*

Metriken oder ein Wechsel des benutzten neuronalen Netzes, falls mehrere zur Verfügung stehen sollten.

Der Anwender hat zudem die Möglichkeit sich diese drei Werte graphisch anzeigen zu lassen. Zu diesem Zwecke wurde in die *WLPrediction*-Anwendung der Open Source Webserver *Jetty* eingebettet. Durch seine schlanke und einfach gehaltene Art bietet er eine effiziente Lösung. Er ist dafür ausgelegt in anderen Anwendungen als Komponente eingebettet zu werden, und da er in Java programmiert wurde erleichterte dies den Einsatz im *WLP-Framework* [Mor08]. Das eigentliche Schaubild, das alle drei Werte als Diagramm abbildet wird mit Hilfe des Java-Pakets *JCharts* erstellt [Kry08]. Diese graphische Darstellung im Browser ist um ein Vielfaches aussagekräftiger als eine Serie gespeicherter Werte in einer Datenbank. Abbildung 2.9 zeigt eine derart erzeugte graphische Darstellung der Vorhersage-Werte. Nachdem nun mit dem Aufbau und den einzelnen Komponenten des *WLP-Frameworks* die Grundlagen für diese Arbeit detailliert betrachtet wurden, soll mit der Weiterentwicklung dieses Frameworks im nächsten Kapitel die eigentliche Hauptaufgabe begonnen werden.

3. Weiterentwicklung des WLP-Frameworks

Im Rahmen dieser Arbeit soll das *WLP-Framework*, das in der Machbarkeitsstudie von Hagmann [Hag07] entstanden ist und als Prototyp vorliegt, entscheidend weiterentwickelt werden. Dieser Prototyp basiert auf den vorliegenden im größten Teil theoretischen Ergebnissen der ersten beiden Arbeiten von Kleeberg [Kle06] und Gebhard [Geb06] des *Joint Research Projects* "Workload Prediction auf Mainframes".

Um einen Einstieg in das komplexe Thema zu finden, besteht die erste Aufgabe darin, eine Art "Bedienungsanleitung" zu erstellen, da in diese Richtung bisher keine Dokumentation vorhanden ist und das *WLP-Framework* sehr umfangreich und komplex in seiner Bedienung ist.

Durch das dadurch angeeignete Wissen wird anschließend die vorhandene *Apache Derby* Datenbank gegen die IBM *DB2 for z/OS* Datenbank getauscht um damit die Umstellung auf das XML-Speicherformat vorzubereiten. Dadurch werden für jeden Zeitstempel sämtliche Metriken in einem eigenen XML-Dokument abgelegt. Diese Umstellung ist der erste Schritt zur Elimination des *generischen Tabellenformats*. Im Anschluss wird die eigentliche Refaktorisierung durchgeführt. Bei diesem weitreichenden strukturellen Umbau werden mehrere Klassen editiert bzw. entfernt um als Ziel die Nachteile des *generischen Tabellenformats* zu beseitigen und das *WLP-Framework* von unnötiger Komplexität zu befreien.

Nach dem die Datenbank schon auf dem Großrechner läuft wird als letztes auch der *RMFRecorder* sowie die *WLPrediction* darauf migriert, indem sie auf den Großrechner übertragen und für die dortige Ausführung konfiguriert werden.

3.1. Erstellen einer Bedienungsanleitung

Das Erstellen einer Bedienungsanleitung ist Teil dieser Diplomarbeit. Da sie stellenweise detaillierte Schritt-für-Schritt Anweisungen mit Screenshots enthält und dadurch einen großen Umfang besitzt, wird sie an dieser Stelle nur kurz angeschnitten Sie befindet sich aber in vollständigem Umfang in Anhang A dieser Arbeit.

Aktuell ist diese Anleitung in sieben Kapitel unterteilt und als "work-in-progress" deklariert. Mit der Weiterführung dieses Projekts sollte auch die Bedienungsanleitung ergänzt werden. Zu Beginn werden die Voraussetzungen für den Betrieb des *WLP-Frameworks* erläutert. Dies ist ein wichtiges Kapitel, da für das erfolgreiche Ausführen der Anwendungen einiges beachtet werden muss. Um neben der Bedienungsanleitung auch eine Art "Checkliste" dieser Voraussetzungen zur Hand zu haben, wird ein Ablaufdiagramm für beide Anwendungen erstellt. Abbildung 3.1 zeigt dieses Diagramm für den *RMFRecorder* und Abbildung 3.2 analog für die *WLPrediction*.

Da für das *WLP-Framework* ein *Vicom-System* zum Einsatz kommt, wird in diesem Kapitel der *Initial Program Load* (IPL) bei Verwendung eines solchen Systems beschrieben. *Vicom* ist ein Virtualisierungssystem, das oftmals beim Entwickeln von Anwendungen für den Großrechner zum Einsatz kommt. Damit ist es beispielsweise möglich für ein System die gewünschte Betriebssystemversion aus einer Liste von vorhandenen Versionen hochzufahren.

Anschließend werden im zweiten Kapitel die Möglichkeiten zur Herstellung einer Verbindung mit der verwendeten Datenbank *DB2 for z/OS* beschrieben. Zu diesem Zweck werden die für das Verbinden nötigen Schritte mit den Programmen *Data Studio Developer* und *SQuirreL* erklärt. Detailliertere Informationen zu diesen beiden Programmen folgen in den Kapiteln 3.2.3.1 und 3.2.3.2.

Im nächsten Kapitel wird die Benutzung des *ABLE-Editors* erläutert. Dabei werden vom Erstellen eines *WlpAgents* über dessen Training bis hin zum abschließenden Speichern im AgentPool sämtliche Schritte im Detail erklärt.

Kapitel vier und fünf behandeln das Starten des *RMFRecorders* und der *WLPrediction*, bevor in Kapitel sechs die Vorgehensweise zur Migration des *WLP-Frameworks* auf den Großrechner angegeben wird.

Das letzte Kapitel umfasst momentan nur eine unsortierte Liste von wichtigen Hinweisen und generellen Tipps zu dem vorliegenden Thema.



Abbildung 3.1.: Ablaufdiagramm für die Inbetriebnahme des RMFRecorder



Abbildung 3.2.: Ablaufdiagramm für die Inbetriebnahme der WLPrediction

3.2. Umstellen der Datenbank von *Apache Derby* auf *DB2 for z/OS*

Obwohl *Apache Derby* eine solide und gut funktionierende Datenbank ist, gibt es mindestens zwei Gründe, die für eine Umstellung auf DB2 sprechen.

Zum einen ist DB2 eine neue Art hybrider Datenbanken, die die Vorteile einer relationalen Datenbank mit denen der XML-Speichertechnologie kombiniert. Dies bietet die Möglichkeit zur Elimination des *generischen Tabellenformats*.

Zum anderen gibt es innerhalb der DB2-Produktfamilie eine als *DB2 for z/OS* bezeichnete Version für den Großrechner. Somit wird durch die Datenbankumstellung gleichzeitig die erste Komponente des *WLP-Frameworks* auf den Großrechner migriert.

3.2.1. DB2 pureXML-Technologie

IBMs DB2 Datenbank wurde 1983 auf den Markt gebracht und hat seither viele Evolutionsstufen hinter sich, wobei in den letzten zwei Dekaden kaum eine so gravierend und herausragend war wie der Sprung auf die Version 9. Neben den üblichen relationalen Daten können in dieser Version auch Dokumente, Audio- und Videodateien, Bilder, Webseiten und digital signierte XML-Transaktionen durch das System verwaltet werden. Laut IBM-Angaben ist DB2 9 branchenweit die erste relationale Datenbank, die neben den oben genannten Daten zusätzlich auch XML nativ speichern kann. Dies wird durch die speziellen XML-Spalten einer relationalen Tabelle realisiert und ermöglicht somit einen nahtlosen Informationsfluss zwischen relationalen Daten und XML-Daten. Diese Eigenschaft wird *pureXML*-Technologie genannt und ist mit ein Grund dafür, dass IBM ihre Datenbanken seither als Daten-Server bezeichnen, die sämtliche Arten von Daten verwalten und zur Verfügung stellen können [IBM06].

Neben der neuen *pureXML*-Technologie wurden zwei weitere signifikante Neuerungen eingeführt, eine als *Venom* bezeichnete Speicherkomprimierung und eine verbesserte autonome Speicherverwaltung. Eine weitere ebenfalls interessante Eigenschaft ist der neuerdings ermöglichte Zugriff auf *Oracle-* und *MySQL-*Datenbanken. Im Gegenzug ist mit Datenbanken dieser beiden Hersteller jedoch kein Zugriff auf DB2 möglich [IBM06]. Da für diese Arbeit nur von der *pureXML*-Technologie intensiv Gebrauch gemacht wurde, beschränkt sich die detailliertere Ausführung nur auf dieses Thema und lässt die anderen Themen außer acht. Für tiefer gehende Informationen zu diesem kontinuierlich in der Entwicklung stehenden IBM-Produkt wird auf [IBM08b] verwiesen.

Laut [IBM07] gab es vor DB2 9 für XML-Daten folgende traditionelle Speichermethoden:

- XML-Dokumente in Dateisystemen speichern
- XML-Daten in relationale Datenbanken als Large Objects (LOBs) speichern
- Aufsplitten der XML-Daten in mehrere relationale Tabellen und Spalten
- Daten isoliert in reinen XML-Datenbanken speichern

Da alle oben genannten Methoden gravierende Nachteile besitzen ist IBM mit DB2 9 einen komplett neuen Weg gegangen. Die relationalen Services und die XML-Services wurden im Datenbank-Kern eng miteinander verzahnt und bilden somit die Grundlage für den branchenweit ersten hybriden Daten-Server für relationale Daten und XML-Daten. Diese neue Art hybrider Datenbanken wird in Abbildung 3.3 veranschaulicht.

Die *pureXML*-Technologie umfasst laut [IBM07] u. a. folgende Eigenschaften:

- Einführung eines neuen *pureXML*-Datentyps und neuer Speichertechniken um hierarchische Strukturen, die in XML üblich sind, effizient zu verwalten
- *pureXML* Index-Technologien um Abfragen auf bestimmte Teile eines XML-Dokumentes zu beschleunigen
- Einführen der standardisierten Abfragesprachen XML Query Language (XQuery) und SQL/XML
- Die Möglichkeit XML-Schemata zu verwalten, zu validieren und weiterzuentwickeln
- Funktionen zum Aufsplitten und Zusammenbauen von XML-Dokumenten aus bestehenden relationalen Daten



Abbildung 3.3.: Hybride Datenbank DB2 9: Kombinierte Speicherung von XML-Daten und relationalen Daten (Abbildung aus [IBM07])

Im Gegensatz zu anderen relationalen Datenbanken, die XML in LOBs speichern, hat DB2 9 extra einen neuen XML-Datentypen eingeführt. Dieser Datentyp speichert die eingelesenen XML-Dokumente in einer hierarchischen Baum-Struktur, wodurch die aufwendige Umwandlung in eine relationale Form entfällt. Durch das Wegfallen dieses Mehraufwands wird das Ablegen und Abfragen von XML-Daten entsprechend effizient ermöglicht. Dieser entscheidende Vorteil wird ebenfalls in der erwähnten Abbildung 3.3 dargestellt.

Mit den in DB2 9 integrierten Möglichkeiten zur hybriden Datenverwaltung erhält man einen führenden Daten-Server für relationale Daten und XML-Daten, der einfach zu bedienen ist, auf Standards basiert und in der Industrie schon jahrelang bewährt ist. Gerade in Zeiten in denen XML in immer mehr Bereichen Einzug hält, ist die effiziente Speicherung und Abfrage ohne zeitaufwendiges Umwandeln in das relationale Modell ein wesentlicher Faktor um Geschäftsprozesse leichter, schneller und billiger zu implementieren.

Im vorliegenden Fall wird ein wesentlicher Nachteil des *WLP-Frameworks*, das *generische Tabellenformat*, mit Hilfe der *pureXML*-Technologie beseitigt.

3.2.2. Durchführung der Datenbankumstellung

Die Umstellung von *Apache Derby* auf *DB2 for z/OS* ist nicht ganz trivial. Die DB2 Version für den Großrechner hat einen historisch gewachsenen Sonderstatus innerhalb der DB2-Produktfamilie. Um die Problematik zu verdeutlichen, müssen zuerst einige Informationen über die DB2-Produktfamilie erläutert werden.

Die wohl wichtigste und älteste DB2-Produktlinie besteht seit 1983 für die Großrechner und wird heute *DB2 for z/OS* genannt. In den 90er-Jahren kamen dann auch Versionen für OS/2, UNIX und Windows dazu. Zu dieser weiteren wichtigen Produktlinie kam inzwischen Linux als Betriebssystem hinzu, und da OS/2 schon lange vom Markt verschwunden ist, wird diese Produktlinie als *DB2 for Linux, UNIX and Windows* oder kurz als *DB2 LUW* bezeichnet. Daneben gibt es noch zwei weitere Produktlinien für *System i* und z/VSE respektive z/VM, die praktisch keine Rolle mehr spielen und auf die deswegen nicht weiter eingegangen wird [Wik08].

Zwischen *DB2 for z/OS* und *DB2 LUW* bestehen entscheidende Unterschiede, angefangen mit der Tatsache, dass Ersteres in PL/S und Letzteres in C++ implementiert wurde.

Zudem ist *DB2 for z/OS* für Kunden mit geschäftskritischen, komplexen und schwierigen Ansprüchen, denen *DB2 LUW* nicht gerecht werden kann weil es darauf nicht ausgelegt ist.

Ein weiterer Unterschied ist die Verknüpfung zwischen Hard- und Software auf dem Großrechner. So besteht bei der Weiterentwicklung von *DB2 for z/OS* als Software wie auch von *System Z* als Hardware eine beidseitige enge Zusammenarbeit. Dies wird beispielsweise dadurch deutlich, dass DB2 auf dem Großrechner ein eigenes Subsystem von z/OS darstellt.

Mit diesem Hintergrund ist es nicht verwunderlich, dass spezielle Eigenschaften, die auf dem Großrechner nicht wegzudenken sind auf *DB2 LUW* nicht existieren und auch gar nicht realisierbar sind, weil die verwendete Hardware dies gar nicht unterstützen bzw. anbieten kann. Als Beispiel soll hier stellvertretend das *Parallel Sysplex Data Sharing* genannt werden, welches exklusiv auf Großrechner beschränkt ist. Dennoch sind beide Versionen in den Kernfunktionen, die ein *Database Management System* (DBMS) ausmachen größtenteils identisch. Die oben aufgeführten Unterschiede stellen nur einen Auszug dar, weitere Unterschiede und allgemeine Informationen zur DB2-Produktfamilie finden sich in [Wan06].

Es soll damit zum Ausdruck gebracht werden, dass DB2 for z/OS in Installati-
on und Administration von vergleichbaren Produkten auf "gewöhnlichen" x86-Architekturen wesentlich abweicht. Da im Rahmen dieser Diplomarbeit eine Einarbeitung in benötigtem Maße nicht möglich war, wurde das Aufsetzen der Datenbank von einem erfahrenen Datenbank-Administrator der IBM durchgeführt. Sämtliche für den Betrieb benötigten Daten und Angaben wurden von diesem nach der Installation zur Verfügung gestellt. Da die Verwendung der *pureXML*-Technologie, welche mit der Version 9 eingeführt wurde, eine zentrale Rolle in dieser Arbeit spielt, muss mindestens diese oder eine aktuellere Version verwendet werden. Im vorliegenden Fall wird *DB2 for z/OS* 9.1 eingesetzt.

Nachdem die Datenbank *WLMD941* angelegt wurde und die Daten für den Host *BOEWLM4* und den Port 5941 bekannt sind, kann mit diesen Daten schließlich die Verbindungs-URL zusammengesetzt werden:

jdbc:db2://BOEWLM4:5941/WLMD941.

Der Austausch der Datenbank ist somit in der Theorie vollzogen. Abschließend müssen dem verwendeten Benutzer *PRAK1* ausreichend Rechte erteilt werden und zudem muss die Umstellung der Verbindungs-URL im Quellcode der Anwendungen *RMFRecorder* und *WLPrediction* erfolgen. Dies wird am besten mit der *Search/Replace*-Funktion von *Eclipse* durchgeführt. Jedes Vorkommen der alten URL *jdbc:derby://localhost:1527/WlpDatabase* muss durch die neue Verbindungs-URL *jdbc:db2://BOEWLM4:5941/WLMD941* ersetzt werden.

Die Datenbankumstellung ist damit erfolgreich abgeschlossen.

3.2.3. Herstellen einer Datenbank-Verbindung

Nachdem die Datenbank aufgesetzt wurde, sollten als nächstes die benötigten Tabellen angelegt werden, da diese zur Laufzeit der Anwendungen *RMFRecorder* und *WLPrediction* schon vorhanden sein müssen. Zu diesem Zweck gibt es je nach Aufgabenbereich und persönlichen Vorlieben viele verschiedene Programme. Im Anschluss wird zum einen der *IBM Data Studio Developer* (im Folgenden als *Data Studio* bezeichnet) und zum anderen die Open Source Anwendung *SQuirreL* betrachtet. Selbstverständlich kann auch jedes andere Programm, welches sich durch einen JDBC-Treiber mit einer Datenbank verbinden lässt verwendet werden, jedoch kann eine reibungslose Funktionsweise nicht gewährleistet werden. Eine weitere Alternative, die hier allerdings nicht näher beschrieben werden soll, bietet die *Interactive* System Productivity Facility(ISPF) mit SQL Processing Using File Input (SPUFI) über eine 3270-Hostsession.

Bezüglich der Verbindung zu *DB2 for z/OS* mit dem JDBC-Treiber muss auf Folgendes hingewiesen werden:

Beim *Data Studio* wird der benötigte Treiber durch die Installation gleich mitgeliefert, dies ist bei *SQuirreL* nicht der Fall. Das liegt daran, dass der Treiber aus lizenzrechtlichen Gründen einzeln nicht frei verfügbar ist und somit auch nicht in beliebige Anwendungen integriert werden kann. Falls *SQuirreL* als SQL-Client verwendet wird, muss entweder das *Data Studio* zusätzlich installiert werden oder der im Verzeichnis DB2z-Treiber der beigelegten DVD bereitgestellte Treiber eingebunden werden. Die Datei db2jcc.jar enthält den Treiber und die Datei db2jcc_licence_cisuz.jar die benötigte Lizenz, wobei beide gleichermaßen benötigt werden.

Informationen zur Konfiguration der Verbindungsherstellung bzw. zum Einbinden des JDBC-Treibers finden sich in der Bedienungsanleitung in Anhang A.

3.2.3.1. IBM Data Studio Developer

Das *Data Studio* bietet eine integrierte Datenbank-Entwicklungsumgebung die auf *Eclipse* basiert und somit ein vertrautes und vielfach bewährtes Werkzeug liefert. Es ist sowohl für Entwickler als auch für Administratoren gedacht und unterstützt beide gleichermaßen mit Funktionen zur Umsetzung von SQL Skripten, *Stored Procedures* oder administrativen Aufgaben [Seu08].

Neben den üblichen Standardfunktionen zum Erstellen und Pflegen von Datenbanken und Tabellen per SQL gibt es für Datenbank-Administratoren laut [Seu08] u. a. folgende Funktionen:

- Erstellen physischer Datendiagramme
- Anzeigen der Datenverteilung
- Objekt-Verwaltung
- Daten-Verwaltung

Für Datenbank-Entwickler hingegen, heben [Seu08] und [IBM08c] u.a. folgende Funktionen hervor:

🗟 Daten - IBM Data Studio Deve	oper			
Datei Bearbeiten Navigieren Sucher	n Projekt Daten Ausführen	Fenster Hilfe		
i 📬 • 🖫 🖻 i 🔓 i 💁 • i	0 🔗 🔁 • 🎱	12-9-6		😭 🔚 Daten
🔁 Datenprojektexpl 🛛 🖵 🗖			- 0	🗄 Gliederung 🛛 🗖 🗖
□ 🔄 🍹				Es ist keine Gliederung verfügbar.
🚺 Datenbank-Explorer 🛛 🗖 🗖	Eigenschaften Tasks Fehler	Fehlerprotokoll Modellberid	cht 0 🖁 Datenausgabe 🙁	Lesezeichen 🗖 🗖
	Status Aktion	Objektname		(SG)
🕒 🔄 🗋 🛓 🚮 🕂 🖕			Nachrichten Parameter E	rgebnisse Daten zur Profilermittlung
····· 🙀 Verbindungen				
			1	

Abbildung 3.4.: Data Studio Startbildschirm

- Entwicklung von Datenbank- und Java-Anwendungen (JDBC)
- Möglichkeit zur schnelleren Entwicklung von Datenbank-Anwendungen mit einem integrierten Abfrage-Editor für SQL und XQuery
- SQL Builder
- XML (Schema-) Editor
- Erstellen und Testen von Stored Procedures (Java und SQL)
- Optimierung von existierenden JDBC-Anwendungen ohne Änderungen am Quellcode

Abbildung 3.4 zeigt einen Screenshot des *Data Studios* in der aktuell vorliegenden Version 1.2.

Durch seine umfassenden Funktionen bietet das *Data Studio* somit viele Vorteile für den längerfristigen Gebrauch. Die Datenbank-Administration sowie die Entwick-

lung und Pflege von Datenbank-Anwendungen werden in großem Maße erleichtert. Häufig verwendete SQL-Befehle können als Skripte abgespeichert und mit einem Klick ausgeführt werden. Außerdem unterstützt es als von IBM entwickeltes Produkt die neue *pureXML*-Technologie am umfassendsten mit zum Teil extra darauf zugeschnittenen Werkzeugen.

Demgegenüber steht aber die Schwerfälligkeit und Komplexität, die solch eine Funktionsfülle mit sich bringt. Es kann einige Zeit dauern bis man sich mit den neuen Funktionen vertraut macht und alles auf die persönlichen Bedürfnisse zugeschnitten wurde. Da es gleichzeitig Entwickler und Administratoren ansprechen soll, wird zudem eine Vielzahl der vorhandenen Funktionen wahrscheinlich nie zum Einsatz kommen.

Bei dieser Arbeit konnte nur auf eine zeitlich beschränkte Demo-Version zurückgegriffen werden, die im Funktionsumfang aber keinerlei Einschränkungen aufwies. Diese kann jederzeit kostenlos von der IBM Webseite herunterladen werden.

3.2.3.2. SQuirreL SQL

Im Gegensatz zum *Data Studio* ist *SQuirreL* eine Open Source Anwendung deren Verwendung kostenlos ist. *SQuirreL* ist ein universeller graphischer SQL-Client für relationale Datenbanken, der auf jede Datenbank zugreifen kann, für die ein JDBC-Treiber verfügbar ist. Außerdem ist es in Java geschrieben und somit auf jeder Javafähigen Plattform lauffähig. Im Vergleich zum *Data Studio* ist *SQuirreL* um ein Vielfaches kleiner und kann wahlweise als installierbare Version oder als gepacktes Archiv bezogen werden. Letzteres liegt sofort nach dem Entpacken in einer lauffähigen Form vor.

Die aktuelle Version ist 2.6.8, wobei sich im Verlauf dieser Arbeit herausgestellt hat, dass in dieser Version eine entscheidende Eigenschaft noch nicht implementiert wurde: Das Anzeigen des nativen XML-Datentyps. Dieses Manko ist in den so genannten *Weekly Snapshots* behoben und erfolgreich getestet worden. Da es stabil läuft kann damit gerechnet werden, dass es in eine der nächsten offiziellen Versionen implementiert wird. Für die vorliegende Arbeit wird einer dieser *Weekly Snapshots* verwendet.

Trotz seiner Kompaktheit bietet *SQuirreL* eine beachtliche Funktionsvielfalt. Der folgende Auszug ist aus [WG08]:

• Betrachten und Editieren von Daten

SQuirreL SQL Client Version 2.6.6	_	
Datei Ireiber Aliase Plugins Session Eenster Hilfe		
Verbinden mit: Wlp 🔻 📽 📰 📰 👘 🕘 Aktive Session: 🔍 👻 🛞	\otimes	C2 H
Treiber 🖶 🥢 🗈 📽 🚝 🚍 Aliase 🖇 🏺 🖉 🗈 🖾 🛱		
X Apache Derby Client		
A Avion		
✓ DB2 for z/OS		
😣 Firebird JayBird		
S FrontBase		
A H2 To Mamony		
Logs: Keine Fehler, Keine Warnung, 25 Nachrichten 📃 📮 🛛 7 von 8 MB 🚺 🗿 🌍 🛛 2	3:00:2	L CET

Abbildung 3.5.: SQuirreL Startbildschirm

- Betrachten von Datenbank-Metadaten
- Paralleles Arbeiten mit mehreren Datenbanken auf beliebigen Servern
- Arbeiten mit verschiedenen Datenbanksysteme über eine einheitlichen Oberfläche
- Einbinden von speziellen Funktionen des jeweiligen Datenbanksystems durch produktspezifische Plugins
- Bearbeiten von Daten wahlweise direkt in Datenbanktabellen oder per SQL-Befehl

In Abbildung 3.5 ist ein Screenshot von SQuirreL zu sehen.

Mit seiner Flexibilität, seiner Leichtigkeit und der Möglichkeit auch heterogene Datenbank-Landschaften mit einem einzigen Werkzeug bearbeiten zu können, bietet *SQuirreL* dem Nutzer gleich mehrere Vorteile.

Für die Entwicklung von Datenbank-Anwendungen ist es hingegen eher ungeeignet und auch in der Administration besitzt es Schwächen, da *SQuirreL* allgemein gehalten wurde, um möglichst viele Datenbanken unterstützen zu können. Dieser Punkt fällt aber kaum ins Gewicht, da die Möglichkeit besteht Plugins zu entwickeln und einzubinden, die spezielle produktspezifische Funktionen implementieren. Durch die aktiven Benutzer und Entwickler dieses Werkzeugs können zu den bereits vorhandenen Plugins jederzeit neue hinzukommen.

Ein weiterer Nachteil sind die benötigten JDBC-Treiber, die man selbst beschaffen und einbinden muss. Aber auch für dieses Problem wurde in Kapitel 3.2.3 eine Lösung vorgestellt. Entweder man installiert das *Data Studio*, da dieses den benötigten Treiber gleich mitinstalliert, oder es wird der auf DVD bereitgestellte Treiber eingesetzt.

Da sich der Zugriff auf die Datenbank in überschaubaren Grenzen hält, wurde im Rahmen dieser Arbeit hauptsächlich *SQuirreL* verwendet. Das *Data Studio* kam nur zu Beginn zum Einsatz um einige Abfragen durchzuführen.

3.3. Elimination des generischen Tabellenformats

Nachdem in den ersten Schritten die Datenbank auf *DB2 for z/OS* 9.1 umgestellt wurde, werden nun das *generische Tabellenformat* und die damit verbundenen Nachteile eliminiert. Aus logischen Gesichtspunkten wird diese Aufgabe im *RMFRecorder* begonnen. Bevor aber mit der Refaktorisierung begonnen werden kann, muss zuerst eine passende Möglichkeit gefunden werden XML mit Java zu verarbeiten. Dies umfasst das Erstellen, Editieren und Einlesen von XML-Dokumenten.

3.3.1. Möglichkeiten der XML-Verarbeitung in Java

Es gibt mittlerweile verschiedene alternative Methoden um XML mit Java zu verarbeiten, je nachdem welches Anwendungsgebiet angesprochen werden soll. Im vorliegenden Fall stellt sich die Frage, wie die Metrik-Daten, die über den CIM-Client abgegriffen werden, für die Weiterverarbeitung in das XML-Format überführt werden können. Dafür gibt es laut [Ull08b] drei verschiedene Verarbeitungstypen:

• DOM-orientierte Schnittstellen (repräsentieren den XML-Baum im Speicher): W3C-DOM, *JDOM*, *dom4j*, *XOM*, ...

- Push-Schnittstellen (nach dem *Callback*-Prinzip ruft der Parser Methoden auf und meldet Element-Vorkommen): *Simple API for XML* (SAX) ist ein populärer Repräsentant
- Pull-Schnittstellen (es wird wie ein *Tokenizer* über die Knoten gegangen): Dazu gehören *XML Pull Parser* (XPP), wie sie die *Streaming API for XML* (StAX) definiert

Da die letzten beiden Schnittstellen ereignisorientiert arbeiten und dadurch das Dokument nur in aktuell verwendeten Stücken geladen wird, eignet sich die Verarbeitung mit Pull- und Push-Schnittstelle eher für Darstellungen von XML-Dateien im Browser oder für die Suche nach bestimmten Inhalten und ähnlichen Aufgaben. Deswegen sind sie für die Lösung des vorliegenden Problems nicht zu empfehlen. Im Gegensatz dazu verfolgt die vom *World Wide Web Consortium* (W3C) standardisierte Schnittstelle des *Document Object Models* (DOM) einen anderen Ansatz: Sie lädt die gesamte XML-Struktur als Baum in den Speicher. Nur so ist die interaktive Verarbeitung eines XML-Dokumentes möglich. Da bei der Datenaufzeichnung im Rahmen dieser Arbeit die interaktive XML-Verarbeitung im Speicher essentiell ist, kommt als Verarbeitungsmöglichkeit nur DOM in Frage.

Ziel der DOM-Spezifikation ist eine Schnittstelle zur Verfügung zu stellen, die es Programmen ermöglicht dynamisch auf ein strukturiertes Dokument zuzugreifen und dessen Inhalt samt Struktur und Stil zu verändern, um es aktualisiert wieder darzustellen. Der Zugriff soll dabei nicht nur möglichst einfach und einheitlich, sondern auch programmiersprachen- und plattformunabhängig erfolgen [Wor08a]. So würde beispielsweise die in Abbildung 3.6 dargestellte XHTML-Tabelle als DOM-Baum der Abbildung 3.7 entsprechen.

Bevor ein bestehendes Dokument verarbeiten werden kann, muss es zuerst durch die Anwendung eingelesen und im Arbeitsspeicher als navigierbarer DOM-Baum angelegt werden. Dann ist es möglich durch das Dokument zu navigieren, Knoten zu erzeugen, löschen und zu verschieben. Es ist auch möglich Attribute und Textinhalte von Knoten auszulesen, zu ändern oder zu löschen. Am Ende der Verarbeitung kann dann aus dem Dokument-Objekt durch Serialisierung ein neues Dokument erstellt werden.

```
Shady Grove
Shady GroveShady Grove
```

Abbildung 3.6.: XHTML-Tabelle (Abbildung aus [Wor08b])



Abbildung 3.7.: Entsprechender DOM-Baum für XHTML-Tabelle aus Abbildung 3.6 (Abbildung aus [Wor08b])

3.3.2. XML-Verarbeitung mit JDOM

Auch bei den Implementierungen der DOM-Schnittstelle für Java sind mehrere Alternativen verfügbar. Betrachtet werden hier *JDOM*, *dom4j* und *XOM*.

Alle drei Projekte sind als Open Source verfügbar, wobei die ersten beiden am weitesten verbreitet sind und *JDOM* die älteste Implementierung darstellt. Das *XOM*-Projekt wurde ursprünglich als Weiterentwicklung von *JDOM* gestartet, bildet mittlerweile aber ein eigenständiges Projekt. Dadurch bestehen mehr Gemeinsamkeiten als Unterschiede zwischen den beiden Implementierungen [Har08]. Diese Tatsache trifft auch auf *dom4j* und *JDOM* zu. Durch die Orientierung an der DOM-Spezifikation unterscheiden sich beide Projekte nur marginal und unwesentlich voneinander [Met08].

Wegen der weiten Verbreitung und der Verwendung samt Tutorial in dem renommierten Werk [Ull08a] fällt die Wahl hier auf die *JDOM*-Schnittstelle. Für die vorliegende Aufgabe werden im Wesentlichen nur die Standard-Funktionen der XML-Verarbeitung benötigt. Da diese Funktionen von allen drei Schnittstellen zur Verfügung gestellt werden, kann die vorliegende Aufgabe genauso gut mit den anderen Schnittstellen gelöst werden.

Durch das Ziel DOM möglichst allgemein zu halten – es muss mit HTML- und XML-Dokumenten arbeiten und mit möglichst allen Browsern kompatibel sein – ist es unhandlich in der Benutzung, ineffizient und vor allem speicherlastig.

JDOM dagegen versucht diese Mängel einzuschränken oder ganz zu beseitigen. Es ist eine Implementierung der DOM-Schnittstelle, die speziell auf Java zugeschnitten ist und somit einige Optimierungen und Verbesserungen beinhaltet. *JDOM* ist nur für XML ausgelegt, setzt *Java Collections* ein und arbeitet praktischerweise mit anderen Standards wie z. B. der *Simple API for XML*(SAX) zusammen.

Als Resultat ist es einfach und sauber implementiert, was wiederum zu einer intuitiveren Schnittstelle führt und eine schnelle fehlerfreie Integration in die eigene Anwendung begünstigt. *JDOM* ist freie Software und daher ein kostengünstiger Einstieg in die XML-Verarbeitung mit Java [JDO08].

Das Erstellen und Bearbeiten eines Dokuments ist denkbar einfach. So sind beispielsweise für das Hinzufügen oder Löschen eines Knotens die Funktionen addContent() und removeContent() zuständig. Wenn bevorzugt mit den Standard Java-Klassen gearbeitet wird, dann können durch die Funktion getChildren() sämtliche Unterknoten als java.util.List übergeben werden, um diese anschließend mit den bekannten Methoden der List-Schnittstelle zu bearbeiten. Alle an der Liste durchgeführten Änderungen werden direkt auf den DOM-Baum im Speicher übertragen. Somit bleibt es dem Entwickler überlassen, wie er das Dokument bearbeiten will.

Neben den üblichen get-Methoden, um durch den Baum zu navigieren, unterstützt *JDOM* auch die ebenfalls vom W3C standardisierte *XML Path Language* (XPath). Dabei handelt es sich um eine Abfragesprache mit der man Teile eines XML-Dokuments adressieren kann [Wor08c]. Die von *JDOM* verwendete XPath-Schnittstelle ist *Jaxen*, eine pure Java-Schnittstelle, die auch auf DOM oder *dom4j* zugeschnitten werden kann [Cod08]. XPath dient auch als Grundlage für eine Reihe weitere Standards, wie z. B. *Extensible Stylesheet Language Transformation* (XSLT), *XML Pointer Language* (XPointer) oder der in Kapitel 3.2.1 im Rahmen der *pureXML*-Technologie kurz erwähnten Abfragesprache XQuery.

XPath betrachtet die XML-Datenstruktur als Baum und mit "/" als XPath-Wurzel. Die zur Navigation verwendete vereinfachte Dateisystem-Notation ist an die Regeln des bekannten UNIX-Dateisystems angelehnt. Analog dazu kann man jeden XML-Knoten von der Wurzel aus adressieren, als ob ein Verzeichnis im UNIX-Dateisystem adressiert wird. Oftmals ist es deutlich einfacher den Pfad zu einem XML-Knoten mit einem XPath-Ausdruck anzugeben als mit einzelnen Methodenaufrufen.

Im Rahmen dieser Arbeit wurden bevorzugt XPath-Ausdrücke eingesetzt.

3.3.3. Refaktorisierung des RMFRecorders

Nachdem die theoretischen Probleme der Refaktorisierung bezüglich XML geklärt wurden, kann nun dazu übergegangen werden sämtliche vom *RMFRecorder* aufgezeichneten Metriken in einer hierarchischen XML-Struktur aufzuzeichnen. Durch diese strukturellen Veränderungen wird aus der ursprünglichen Tabelle metrics aus Kapitel 2.2 die Tabelle metrics_xml mit den Spalten *Timestamp* und *MetricsXML*. Der SQL-Befehl für diese neue auf XML umgestellte Tabelle sieht wie folgt aus:

```
CREATE TABLE metrics_xml
(
   Timestamp TIMESTAMP NOT NULL,
   MetricsXML XML NOT NULL,
   PRIMARY KEY (Timestamp)
);
```

Durch die neue XML-Spalte ist auch eine Anpassung des Index notwendig. DB2 *pureXML* führt dazu eine neue Index-Technologie ein, um die Abfrage auf XML-Dokumenten zu beschleunigen. Dazu muss zum gewöhnlichen SQL-Befehl CREATE INDEX zusätzlich noch ein so genanntes *xmlpattern* angegeben werden. Dieses *xmlpattern* besteht aus einem XPath-Ausdruck, welcher eine Teilmenge eines XML-Dokuments spezifiziert [IBM07]. Der Index im vorliegenden Fall soll auf die XML-Knoten *Instance* und *Metric* angewendet werden, genauer auf das Attribut *ID* dieser Knoten. Im Gegensatz zur alten Tabelle muss der Index auf die Instanz-ID einer Metrik explizit angelegt werden. Dort war die Instanz-ID Teil des Schlüssels, wodurch automatisch ein Index darauf angelegt wurde. Da zusätzlich auch noch die Metrik-IDs von Interesse sein könnten, wird mit den folgenden SQL-Befehlen auf beide XML-Teilmengen ein Index angelegt:

```
CREATE INDEX instID_idx on metrics_xml(metricsXML)
GENERATE KEY USING XMLPATTERN
'/MonitoredMetrics/Metric/Instance/@ID'
AS SQL VARCHAR(50);
CREATE INDEX metrID_idx on metrics_xml(metricsXML)
GENERATE KEY USING XMLPATTERN
'/MonitoredMetrics/Metric/@ID'
AS SQL VARCHAR(50);
```

Durch die neue Tabellen-Struktur mit nur zwei Spalten existiert für jeden Zeitstempel genau ein Tupel in der Tabelle und nicht wie zuvor für jede aufgezeichnete Instanz-ID noch ein weiteres zusätzlich Tupel pro Zeitstempel. Diese Tatsache ist der Schlüssel zur Elimination des *generischen Tabellenformats*. Jedes Tupel wird eindeutig durch einen Zeitstempel definiert und somit ist die Reihenfolge in der Tabelle garantiert und eindeutig. Dies konnte im alten Prototypen des *WLP-Frameworks*

1 🍪 🗶 😩 🖞 🖉 🖗	L 🖻 🗶 🖗 🖗	i q 🕹 🤞	i 💀 🕅) f0		
bjects \ SQL \ Hibernate \						
WLPDB1	Imported Keys \ Inde		Column Privil	eges Row IDs	Versions	
🕀 🗠 🛄 ADB	Source) Info Co	ntent Dow Co	ount \ Colu	mps Primary	Key Ex	ported Key
DB2OSC	Source (Into do	Rone Row Co			Key (LA	Jonted Key
DB2OSCA	TIMESTAMP	INSTANCEID	METRICID	METRICVALUE	DATATYPE	COLNUM
H DSNACC	1970-01-01 02:00:00.0	LPI_BATCH1A.1	LPI	10.97	5	2
	1970-01-01 02:00:00.0	LPI_BATCH2A.1	LPI	1.85	5	4
PRAK1	1970-01-01 02:00:00.0	LPI_BATCH2B.1	LPI	1.91	5	5
	1970-01-01 02:00:00.0	LPI_BATCH3A.1	LPI	6.7	5	0
	1970-01-01 02:00:00.0	LPI_BATCH4A.1	LPI	8.2	5	1
E GLOBAL TEMPORARY TABLE	1970-01-01 02:00:00.0	LPI_BATCH5.1	LPI	0.0	5	7
MATERIALIZED QUERY TABLE	1970-01-01 02:00:00.0	LPI_CICS2A.1	LPI	0.0	5	8
E SYNONYM	1970-01-01 02:00:00.0	LPI_CICS3B.1	LPI	0.0	5	9
E SYSTEM TABLE	1970-01-01 02:00:00.0	LPI_TSO1.1	LPI	0.01	5	6
	1970-01-01 02:00:00.0	LPI_USS.1	LPI	3.0	5	3
HETRICS (94500) HETRICS HETRICS	1970-01-01 02:08:20.0	LPI_BATCH1A.1	LPI	4.83	5	2
E METRICS_XML (12)	1970-01-01 02:08:20.0	LPI_BATCH2A.1	LPI	0.77	5	4
E PREDICTION (2505)	1970-01-01 02:08:20.0	LPI_BATCH2B.1	LPI	0.96	5	5
T060619A_D55 (8350)	1970-01-01 02:08:20.0	LPI_BATCH3A.1	LPI	3.08	5	0
T060619A_DS5_XML (835)	1970-01-01 02:08:20.0	LPI_BATCH4A.1	LPI	4.09	5	1
🕀 🗀 VIEW	1970-01-01 02:08:20.0	LPI_BATCH5.1	LPI	0.0	5	7
	1970-01-01 02:08:20.0	LPI_CICS2A.1	LPI	0.0	5	8
	1970-01-01 02:08:20.0	LPI_CICS3B.1	LPI	0.0	5	9
🕀 🙆 SEQUENCE	1970-01-01 02:08:20.0	LPI TSO1.1	LPI	0.0	5	6
	1970-01-01 02:08:20.0	LPI USS.1	LPI	1.98	5	3
🗄 🗀 SYSIBM	1970-01-01 02:16:40.0	LPI BATCH1A.1	LPI	11.12	5	2
E SYSIBMTS	1970-01-01 02:16:40.0	LPI BATCH2A.1	LPI	0.81	5	4
🗄 🗀 SYSFUN	1970-01-01 02:16:40.0	LPI BATCH2B.1	LPI	0.92	5	5
	1970-01-01 02:16:40.0	LPI BATCH3A.1	LPI	2.69	5	0
	1970-01-01 02:16:40.0	LPI BATCH4A.1	LPI	6.32	5	1
	1970-01-01 02:16:40.0	LPI BATCH5.1	LPI	0.0	5	7
	1970-01-01 02:16:40.0	LPI CICS2A.1	LPI	0.0	5	8
	1970-01-01 02:16:40.0	LPI_CICS3B.1	LPI	0.0	5	9
	1970-01-01 02:16:40.0	LPI_TSO1.1	LPI	0.0	5	6
	1970-01-01 02:16:40 0	I DT LISS 1	IDT	0.7	5	3

Abbildung 3.8.: Das alte generische Tabellenformat (Screenshot aus SQuirreL)

nicht garantiert werden, ist aber für die Verwendung von künstlichen neuronalen Netzwerken zwingend erforderlich, wie in Kapitel 2.2 ausführlich beschrieben wurde. Durch die Elimination des alten Tabellenformats ist es möglich das komplette *WLP-Framework* entscheidend zu vereinfachen, da das bis dato verwendete komplizierte Hilfskonstrukt damit überflüssig geworden ist.

Abbildungen 3.8 und 3.9 zeigen Screenshots aus *SQuirreL*, jeweils mit altem und neuem Tabellenformat.

Nach der Anpassung der Tabellen müssen als nächstes die durch den CIM-Client bereitgestellten Daten mit Hilfe von *JDOM* zu einem XML-Dokument zusammengebaut werden. Dazu wird in der Klasse Database.java die Funktion insertRecords() editiert. Dabei muss beachtet werden, dass jede Metrik-ID auch mehrere Instanz-IDs haben kann. Deswegen wird ein Algorithmus implementiert, der für jede Metrik-ID erst einmal prüft, ob diese im *JDOM*-Baum bereits exis-

🧻 1 - WLPDB1 as prak1								
<i>₽ 💠 ★ 幹</i> Ľœħ ₽ ₽	. 🖻 🗶 🗘 🖯	5 o, el 🧔 🖣 🕅 🖆						
Objects \SQL \ Hibernate \								
E WLPDB1 Imported Keys \ Indexes \ Privileges \ Column Privileges \ Row IDs \ Versions \								
E ADB	Source) Info Co	Intent Pow Count Columns Primary Key Exported Keys						
E DB2OSC	Jource / Into Co	Row Council Columns (Primary Key (Exported Keys)						
DB2OSCA	TIMESTAMP	METRICSXML						
DSNACC	1970-01-01 02:00:00.0	<pre><monitoredmetrics><metric id="LPI"><instance id="LPI_BATCH3</pre></td></tr><tr><td></td><td>1970-01-01 02:08:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>PRAK1</td><td>1970-01-01 02:16:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 02:25:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>H AUXILIARY TABLE</td><td>1970-01-01 02:33:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 02:41:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 02:50:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 02:58:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 03:06:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 03:15:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>HIME METRICS (94500)</td><td>1970-01-01 03:23:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>HIME METRICS_XML (12)</td><td>1970-01-01 03:31:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>PREDICTION (2505)</td><td>1970-01-01 03:40:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>1060619A_D55 (8350)</td><td>1970-01-01 03:48:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>H 2 1060619A_D55_XML (835)</td><td>1970-01-01 03:56:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 04:05:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 04:13:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 04:21:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 04:30:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 04:38:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td>H SYSIBM</td><td>1970-01-01 04:46:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 04:55:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:03:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:11:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:20:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:28:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:36:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:45:00.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 05:53:20.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3</td></tr><tr><td></td><td>1970-01-01 06:01:40.0</td><td><MonitoredMetrics><Metric ID=" lpi"=""><Instance ID="LPI_BATCH3</td></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></metric></monitoredmetrics></pre>						
/WLPDB1/PRAK1/TABLE/T060619A_DS5_XML (8	35)	1,1						

Abbildung 3.9.: Das neue XML-Tabellenformat (Screenshot aus SQuirreL)

Z C:\	Docu	ments and Settings\Administrator\Desktop\MonMetrics.xml - Notepad++ 📃 🗆 🔀
File E	dit S	zarch View Format Language Settings Macro Run TextFX Plugins Window ? X
	B	
📄 Mo	onMetri	cs xml
	1	<pre><?xml version="1.0" encoding="UTF-8"?></pre>
	2 🗄	<pre><monitoredmetrics></monitoredmetrics></pre>
	3 [<pre>A <metric id="8D0D50#wT"></metric></pre>
	8 [<pre>Metric ID="8D0D50"></pre>
1	3 [<pre><metric id="8D39C0"></metric></pre>
1	8 8	<pre></pre>
2	3 [<pre><metric id="8D0420"></metric></pre>
2	8 -	<pre><metric id="8D17E0#scp"></metric></pre>
2	9 [<pre><instance id="8D17E0#scpSYSSTC1.1"></instance></pre>
3	0	<metricvalue>0.0</metricvalue>
3	1	
3	2 -	<pre><instance id="8D17E0#sopSYSSTC2.1"></instance></pre>
3	5 8	<pre><instance id="8D17E0#scpSYSOTHER.1"></instance></pre>
3	8 8	<pre><instance id="8D17E0#scpSYSSTC.1"></instance></pre>
4	1 [<pre>Instance ID="8D17E0#scpSYSSTC3.1"></pre>
4	4 -	<pre>Instance ID="8D17E0#sopSYSTEM.1"></pre>
4	7 8	<pre><instance id="8D17E0#scpUSS.1"></instance></pre>
5	0 8	<pre>Instance ID="8D17E0#sopSYSSTC4.1"></pre>
5	3 [<pre>Instance ID="8D17E0#scpSYSSTC5.1"></pre>
5	6	
5	7 8	<pre><metric id="8D3740#scp"></metric></pre>
8	6 6	<pre><metric id="8D3790#scp"></metric></pre>
11	5 [<pre><metric id="8D2D00#scp"></metric></pre>
14	4 -	<pre><metric id="8D35D0#scp"></metric></pre>
17	з Е	<pre><metric id="8D2D40#scp"></metric></pre>
20	2	
eXtensib	ole Mar	kup Language file nb char : 6261 Ln : 201 Col : 12 Sel : 0 Dos\Windows ANSI INS

Abbildung 3.10.: XML-Dokument mit aufgezeichneten Metriken (erstellt mit Hilfe von *JDOM*)

tiert. Ist dies nicht der Fall, wird mit der Funktion buildMetricNode() ein neuer XML-Knoten mit dieser Metrik-ID und den entsprechenden Unterknoten angelegt. Falls diese jedoch schon existiert, werden nur die Unterknoten an den schon existierenden Knoten angehängt. Ein Unterknoten umfasst in diesem Fall immer eine Instanz-ID und den eigentlich gesuchten Wert *MetricValue*. Ein so erstelltes XML-Dokument wird in Abbildung 3.10 dargestellt.

Somit werden alle Metriken in einem XML-Dokument pro Zeitstempel gespeichert und der erste Schritt zur Elimination des *generischen Tabellenformats* ist gemacht.

3.3.4. Refaktorisierung der WLPrediction

Um die Qualität der Vorhersage-Ergebnisse vergleichen zu können, muss auf die in [Geb06] generierten Metrik-Daten zurückgegriffen werden. Bevor al-

so die Vorhersage-Anwendung refaktorisiert werden kann, sollten als erstes diese Referenzdaten in das neue XML-Speicherformat überführt werden. Zu diesem Zweck ist das Konvertierungsprogramm *Converter* entstanden. Es ist als Teil des *WLPrediction*-Projekts angelegt und besteht nur aus der Klasse com.ibm.wlp.converter. Die einzige Aufgabe dieses Programms ist die einmalige Konvertierung von relational gespeicherten Metrik-Daten in das notwendige XML-Speicherformat. Dazu sind folgende Angaben nötig: Zugangsdaten zur Datenbank, die zu konvertierende Tabelle sowie die zu befüllende XML-Tabelle. Im Fall der verwendeten Referenzdaten sind dies die Tabellen T060619A_DS5 und T060619A_DS5_XML. Die 8350 Referenzdatensätze reduzieren sich um den Faktor 10 auf 835, da in der ursprünglichen Tabelle für jeden Zeitstempel 10 Tupel existieren während nach der Konvertierung jeweils nur ein einziger übrig bleibt.

Dieses Konvertierungsprogramm erhebt keinerlei Ansprüche auf korrekte Funktionsweise, es ist als provisorische Lösung anzusehen. Allerdings konnte die Konvertierung der vorliegenden Daten damit erfolgreich durchgeführt werden.

Bei der Refaktorisierung der *WLPrediction* wird systematisch an das Problem herangegangen und bei der ersten Funktion angesetzt, welche das Tabellenformat involviert. Um außerdem das Fehlerrisiko zu verringern, werden die alten Funktionen und die alte Datenstruktur parallel zu der neuen beibehalten und erst nach erfolgreicher Umstellung auskommentiert. Nach jedem Schritt der Umstellung wird die Anwendung erst einmal ausführlich getestet und die Vorhersage-Ergebnisse mit den Ergebnissen der Referenzdaten verglichen, um eventuelle Fehler zu lokalisieren und sofort zu beseitigen bevor sie sich ansammeln. Erst wenn garantiert ist, dass die Anwendung fehlerfrei durchläuft und die gleichen Ergebnisse wie bei den Referenzdaten ausgegeben werden, wird der alte Quellcode komplett auskommentiert, alte Verweise angepasst oder entfernt und unnötige *Import*-Anweisungen oder ähnliche Spuren beseitigt. Nachdem dies erledigt wurde, kann der nächste Vereinfachungsschritt begonnen werden.

Die oben erwähnten ausführlichen Tests werden nach dem folgenden Schema durchgeführt:

Als erstes wird der *ABLE-Editor* im Debug-Modus von *Eclipse* mit dem Ziel gestartet einen neuen *WlpAgent* anzulegen. Der eigentliche Test beginnt dann mit einem Klick auf den Button *Generate Beans* und dem damit verbundenen Initialisieren des *WlpAgents* (vgl. Anleitung in Anhang A).

Dabei muss unbedingt darauf geachtet werden, dass nach jeder Änderung im

Quellcode das Projekt *WLPrediction* neu erstellt wird. Der Grund liegt darin, dass der *ABLE-Editor* auf Ressourcen des *WLPrediction*-Projekts zugreift. Dies wiederum sind diejenigen Ressourcen, die refaktorisiert werden. Mit dem Ausführen des *Ant*-Skripts build.xml im *WLPrediction*-Projekt werden die im Code vorgenommenen Änderungen kompiliert und in die *jar*-Dateien wlpcore.jar und wlprediction.jar gepackt. Wenn im Anschluss der *ABLE-Editor* gestartet wird, dann greift er auf die aktualisierten Funktionen in diesen zwei *jar*-Dateien zu und die vorgenommenen Änderungen können validiert werden.

Im Prototypen von Hagmann [Hag07] erfolgt die Konvertierung vom *generischen* in das für die künstlichen neuronalen Netze benötigte relationale Format in der Klasse TableWrapper. Diese greift wiederum auf den speziellen Datentyp einer assoziativen Tabelle AssocTable zu, welcher eigens für den Prototypen erstellt wurde. Um die zwei Elemente TableWrapper und AssocTable wurde das umfangrei-

che Hilfskonstrukt gebaut, welches am Ende entfernt wird.

Die Refaktorisierung wird mit dem TableWrapper begonnen. Dazu wird das Laden der zu verarbeiteten Daten im Debug-Modus schrittweise verfolgt, um genau festzustellen wann welche Funktion auf die TableWrapper-Klasse zugreift. TableWrapper ist als Superklasse von SeqDBTable angelegt und Letztere wiederum ist eine an das *generische Tabellenformat* angepasste Version der AbleDBTable-Klasse aus dem originalen *ABLE-Framework*.

Das Präfix Seq in den Klassen-Namen soll darauf hinweisen, dass die Klasse an das *generische Tabellenformat* mit seiner sequentiellen Tabellen-Struktur angepasst wurde. Als Ziel für die Refaktorisierung kann folglich festgehalten werden, dass sämtliche Klassen die ein Seq im Namen tragen an das neue Format angepasst werden müssen. Zum Beseitigen der Klasse TableWrapper wird diese zunächst mit allen Funktionen in die Klasse SeqDBTable verschoben, um sie anschließend an das neue XML-Tabellenformat anzupassen und wieder in AbleDBTable umzubenennen.

Als nächstes wird die Datenstruktur der assoziativen Tabelle ersetzt. Dafür muss zunächst geklärt werden, wie die aus der Datenbank ausgelesene Tabelle der Metrik-Daten java-intern verarbeitet werden soll. In Java gibt es aktuell keinen Datentypen und keine Datenstruktur, die explizit für das Abbilden einer Tabelle konzipiert wurde. In Microsofts .*NET* beispielsweise gibt es für solche Fälle die Datentypen DataTable, DataColumn und DataRow, um entweder eine komplette Tabelle, ihre Spalten oder ihre Zeilen in eine Anwendung zu laden. Eine gängige Lösung um Tabellen in Java zu implementieren ist ein zweidimensionaler Array. Da dies die einfachste Methode ist und sie für die vorliegenden Aufgaben ausreichend ist, wird sie hier als Lösung gewählt. Dadurch kann verhältnismäßig einfach eine Tabellen-Struktur mit Spalten und Zeilen abgebildet werden, und so über Ansteuerung der einzelnen Elemente mit Hilfe zweidimensionaler Koordinaten eine transparente Verarbeitung der Daten erreicht werden. Während der Umsetzung hat sich der Datentyp in manchen Fällen aber als zu "starr" erwiesen, weil beispielsweise nach der Initialisierung eines Arrays seine Größe nicht mehr verändert werden kann. So ist in der Folge eine bei Tabellen oft gebrauchte Aktion wie das Entfernen einer Spalte oder Zeile nicht ohne weiteres möglich. Da diese Arbeit eine reine Machbarkeitsstudie darstellt und eine optimale Implementierung nur zweitrangig ist, kann dieser kleine Makel vernachlässigt werden.

Bei der Ablösung der AssocTable-Datenstruktur wird systematisch nach jedem Vorkommen dieser Datenstruktur gesucht, parallel dazu an der selben Stelle die neue aufgebaut und die Werte von alter und neuer Datenstruktur auf Gleichheit überprüft. Beim Auftreten eines Fehlers kann somit genau verfolgt werden ab welchem Punkt die Werte voneinander abweichen.

Von diesem Austausch ist nicht nur die Klasse SeqDBTable betroffen, sondern auch MyDataSetDefinition, eine angepasste Version der Klasse AbleDataSetDefinition aus dem originalen *ABLE-Framework*, welche die von ABLE benötigten Metadaten der SQL-Abfrage liefert.

Größere Probleme treten erstmals bei der Funktion validate() in SeqDBTable auf. Diese Funktion regelt die Vorgehensweise für Ausreißer, falsch skalierte und fehlende Werte einer für die Vorhersage zu verwendenden Instanz-ID. Vor allem im Fall einer kompletten Streichung der Instanz-ID, stößt die Array-Datenstruktur auf oben schon angesprochene Grenzen: Die betroffene Instanz-ID, die intern als Spalte des zweidimensionalen Arrays implementiert ist, kann nicht komplett entfernt werden, da der Java-Array in seiner Größe nicht verändert werden kann. Dieses Problem ist aber mit relativ wenig Aufwand über die Benutzung eines zweiten temporären Arrays lösbar.

Nachdem die neue Struktur vollständig parallel zur alten implementiert ist und auch die Vorhersagen mit beiden Strukturen in unterschiedlichen Datenbank-Tabellen gleiche Werte zurückschreiben, kann die alte Datenstruktur schließlich auskommentiert und alles auf die neue Struktur zugeschnitten werden.

Bei der Überprüfung der graphischen Darstellung der Ergebnisse mit Hilfe von

Jetty, taucht dann ein weiteres Problem auf. Die Funktion basicChart() in der Klasse ImagesServlet, die für das Darstellen des Diagramms zuständig ist, funktioniert nicht fehlerfrei. Die Vorhersage beginnt nicht wie erwartet um die Anzahl von Vorhersageschritten verzögert, sondern zum Zeitpunkt Null.

In der alten Datenstruktur wurden die Vorhersageschritte im *WlpAgent* festgelegt und über das komplizierte Hilfskonstrukt der Funktion basicChart() bereitgestellt. Obwohl diese Information weiterhin im *WlpAgent* festgelegt wird, steht sie durch die Beseitigung des Hilfskonstrukts der Funktion basicChart() nicht mehr zur Verfügung. Deswegen wird mit relativ wenig Aufwand ein Algorithmus implementiert, der den gewünschten Wert ermittelt und der Funktion bereitstellt.

Nachdem nun auch die Vorhersage-Ergebnisse korrekt sind, muss als letzter Schritt der Quellcode "aufgeräumt" werden. Sämtliche Klassen mit dem Präfix Seq werden in ihre Original-Namen aus dem *ABLE-Framework* umbenannt. So wird beispielsweise aus SeqDBTable wieder AbleDBTable und aus SeqDBImportImpl wird AbleDBImportImpl. Im Anschluss müssen auch sämtliche Verweise darauf angepasst oder falls nicht mehr benötigt entfernt werden.

Im Rahmen dieser umfangreichen Refaktorisierung werden zahlreiche Funktionen in verschiedenen Klassen angepasst, in der SeqDBTable-Klasse z.B. die Funktionen getRowCount(), getColumnCount(), getValueAt(), getColumnName(), getRowKey(), isLoaded(), loadDataFile(), validate(). Um die erfolgten Änderungen auch zu einem späteren Zeitpunkt noch nachvollziehen zu können, werden die ursprünglichen von Hagmann [Hag07] erstellten Klassen nicht aus dem Eclipse-Workspace entfernt, sondern nur vollständig auskommentiert und mit einem Unterstrich als erstes Zeichen im Namen markiert. Dabei handelt es sich um die folgenden RemoteSeqDBImport, SeqDBImport, SeqDBImportBeanInfo, Klassen: SeqDBImportCustomizer, SeqDBImportDataBean, SeqDBImportImpl, SeqDBImportPanel, SeqDBImportProxy, SeqDBTable, TableWrapper, AssocLinkedList, AssocListMap und AssocTable.

Abgesehen davon wird die aufwendige Refaktorisierung genutzt, um allgemein den Quellcode zu optimieren und Fehler zu bereinigen. So werden die Start-Skripte angepasst, Ausgaben auf die Konsole kosmetisch und funktional optimiert, unnötige Funktionen wie das in der *WLPrediction* nicht mehr verwendete *Downsampling* entfernt, und wo nötig der Quellcode kommentiert. Als konkretes Beispiel soll hier die Benennung der drei Ausgabewerte der *WLPrediction*-Anwendung angeführt werden. Da die bisherige Benennung des geglätteten (*LPI_BATCH2A.1_target:WlpAgent*) und des vorhergesagten Wertes (*LPI_BATCH2A.1:WlpAgent*) recht verwirrend ist, wurde die Bezeichnung für den Vorhersagewert in *LPI_BATCH2A.1_prediction:WlpAgent* abgeändert.

3.4. Migration des *WLP-Frameworks* auf den Großrechner

Mit Abschluss der letzten Aufgabe dieser Arbeit soll es möglich sein, das *WLP-Framework* komplett vom Client loszulösen und auf den Großrechner zu migrieren. Im ersten Schritt wurde die Datenbank von *Apache Derby* auf *DB2 for z/OS* umgestellt und läuft seither auf dem Großrechner. Die beiden anderen Komponenten, *RMFRecorder* und *WLPrediction*, müssen für die Migration auf den Großrechner erst vorbereitet werden. Im Verzeichnis des *WLPrediction*-Projekts befindet sich ein Unterverzeichnis mit dem Namen dist, welches alle zur Ausführung benötigten Dateien enthält. Eine Übersicht über den Inhalt dieses Verzeichnisses findet sich in Abbildung 3.11.

Im Unterverzeichnis lib befinden sich alle zur Ausführung benötigten Dateien, darunter die für ABLE benötigten *jar*-Dateien, der JDBC-Treiber und alle anderen beim Start der Anwendungen referenzierte *jar*-Dateien.

In einem weiteren Unterverzeichnis mit dem Namen AgentPool befinden sich die erstellten und trainierten Agenten, die jederzeit erweitert werden können.

Die eigentlichen Anwendungen befinden sich im Verzeichnis wlp. Bei Änderungen und Weiterentwicklungen am Quellcode muss jedes Mal durch Ausführen des *Ant-Skripts* build.xml neu kompiliert werden. Dieses Skript kopiert dann automatisch die neue Version der Anwendungen in das dist-Verzeichnis an die entsprechende Stelle.

Im dist-Verzeichnis befinden sich auch die Konfigurationsdateien für das WLP-Framework: WLPDefaults.properties für die allgemeine Konfiguration und cimmetrics.xml mit zugehöriger Schema-Datei cimmetrics.xsd für die Merkmalsselektion. Die für die Ausführung der beiden Anwendungen benötigten Shell-Skripte befinden sich im selben Verzeichnis. Für den RMFRecorder ist das die Datei startRMFRecorder und für die WLPrediction die Datei startWLP.

🖻 dist								
File Edit View Favorites Tools Help								
Search Polders								
🕴 Address 🗁 C: \Documents and Settings \Administrator \My Documents \Diplomarbeit \workspace \WLPrediction \dist 🛛 💽 Go								
	Name 🔺	Size	Туре	Date Modified				
File and Folder Tasks 🛛 💐	C AgentPool		File Folder	13.12.2008 00:31				
	ib 🔁		File Folder	23.09.2008 19:26				
Other Places 🛛 🗧 😵	🗀 wlp		File Folder	22.10.2008 14:23				
	🔮 cimmetrics.xml	2 KB	XML Document	11.12.2008 17:47				
Details 😵	immetrics.xsd	2 KB	XSD File	03.11.2008 17:15				
	🖾 startRMFRecorder	1 KB	File	03.11.2008 14:49				
	startRMFRecorder.bat	1 KB	MS-DOS Batch File	03.11.2008 14:05				
	🖾 startWLP	2 KB	File	03.11.2008 14:46				
	startWLP.bat	1 KB	MS-DOS Batch File	03.11.2008 14:05				
	WLPDefaults.properties	2 KB	PROPERTIES File	12.12.2008 13:58				
10 objects	10 objects 7,67 KB 🚽 My Computer							

Abbildung 3.11.: Übersicht über den Inhalt des dist-Verzeichnisses

Das dist-Verzeichnis muss komplett mit Inhalt auf den Großrechner übertragen werden. Gegebenenfalls muss es nach vorgenommenen Änderungen am Quellcode erneut übertragen werden. Dies ist an sich eine triviale Aufgabe, die aber bei mangelnder Erfahrung und wegen der speziellen Architektur der Großrechner dennoch Probleme bereiten kann.

Erstens verwenden Großrechner die spezielle Zeichenkodierung *Extended Binary Coded Decimals Interchange Code* (EBCDIC). Deswegen muss bei der Übertragung der Dateien auf den Großrechner darauf geachtet werden, welches Format die jeweilige Datei besitzt. Gewöhnliche Text-Dateien, dazu gehören auch HTMLund Skript-Dateien, müssen im ASCII-Modus übertragen werden. Alle anderen Dateien, die proprietäre Dateiformatierungen besitzen, müssen im Binär-Modus übertragen werden. Zu Letzteren gehören beispielsweise *doc-, exe-, zip-* und *jar-*Dateien.

Die Umwandlung von ASCII zu EBCDIC geschieht während der Übertragung automatisch im Hintergrund. Es wird dabei nicht wie sonst bei einer ASCII-Übertragung üblich nur eine Konvertierung zwischen den Zeilenumbruchsvarianten der verschiedenen Betriebssystemen vorgenommen, sondern eine "echte" Konvertierung der Zeichenkodierung. Diese Übertragung findet üblicherweise per *File Transfer Protocol* (FTP) statt, wofür ein FTP-Client benötigt wird. Im Internet gibt es zahlreiche dieser Clients oftmals als kostenlose Software, z.B. *Filezilla*, *WinSCP*, oder auch das *Firefox*-Plugin *FireFTP*.

Interessanterweise müssen die XML-Dateien cimmetrics.xml und cimmetrics.xsd im binären Modus übertragen werden, auch wenn sie nach der Übertragung auf dem Großrechner nicht korrekt darstellbar sind, da ansonsten der *RMFRecorder* nicht funktioniert. Dies kann damit zusammenhängen, dass das Projekt auf dem Client für ASCII kompiliert wird und der *RMFRecorder* deswegen auch eine in ASCII kodierte Datei zum Einlesen erwartet. Dies ist ein unerwartetes Verhalten und sollte in anschließenden Arbeiten nach Möglichkeit genauer untersucht werden.

Zweitens muss geklärt werden wohin die Dateien übertragen werden müssen. Auf Großrechnern gibt es ein extra Subsystem, die *UNIX System Services* (USS), die es ermöglichen UNIX-Anwendungen auf dem Großrechner auszuführen. Dies gilt auch für Java-Anwendungen wie im vorliegenden Fall. Wie bei UNIX üblich müssen die Dateien in das jeweilige Benutzerverzeichnis kopiert werden, da man nur für diese Destination ausreichend Rechte besitzt. In der Regel ist dies das Verzeichnis /u/"Benutzername". Ausführlichere Details zur Datei-Übertragung auf den Großrechner finden sich in der angehängten Anleitung in Anhang A.

Drittens muss darauf geachtet werden, welche Java-Version benutzt wird. Auf dem hier verwendeten System *BOEWLM4* ist die *Java Runtime Environment* (JRE) in Version 1.5 installiert. Eine Kompilierung des Projekts mit Java 1.6 ist auf dem System somit nicht lauffähig und muss zwingend mit einer älteren Java-Version kompiliert werden.

Schließlich müssen nach erfolgreicher Übertragung die entsprechenden Dateirechte mit dem UNIX-Befehl chmod gesetzt werden, um die Anwendung ausführen zu können. Dazu reicht es wenn die *Shell-Skripte*, die zum Starten der Anwendungen erstellt wurden mit den benötigten Rechten ausgestattet werden. Für den *RMFRecorder* ist dies die Datei startRMFRecorder und für die *WLPrediction* ist es die Datei startWLP im Verzeichnis dist.

Abschließend soll hier bezüglich der möglichen Fehlerquellen nochmals auf die Bedienungsanleitung in Anhang A verwiesen werden. Zu sämtlichen oben aufgeführten Punkten finden sich dort detailliertere Hilfestellungen.

4. Ergebnisse

Dieses Kapitel dient der Auflistung, der im Rahmen dieser Arbeit erbrachten Ergebnisse. Dabei wurden zu Beginn vier Aufgaben festgelegt, zu denen nun die jeweiligen Ergebnisse präsentiert werden.

Zur generellen Erleichterung der Bedienbarkeit wurde damit begonnen eine Bedienungsanleitung zu erstellen, die kontinuierlich ergänzt wurde. Im nächsten Schritt erfolgte die Umstellung der Datenbank von *Apache Derby* auf *DB2 for z/OS*, bevor in einem weiteren Schritt das bisher verwendete *generische Tabellenformat* eliminiert wurde. Schließlich erfolgte noch die Migration des *WLP-Frameworks* vom Windows-Client auf den Großrechner.

4.1. Bedienungsanleitung

Als erstes Ergebnis ist eine ausführliche Bedienungsanleitung für das *WLP-Framework* entstanden, welche der Diplomarbeit angehängt wurde und in Anhang A zu finden ist. Bisher war keinerlei Dokumentation vorhanden, aber aufgrund der Komplexität des *WLP-Frameworks* erschien eine solche Bedienungsanleitung notwendig und sinnvoll. Sie stellt eine Art Sammlung aller aufgetretenen Erfahrungen und Probleme beim Betrieb des *WLP-Frameworks* dar. Dennoch ist diese Bedienungsanleitung nicht vollständig und sollte in weiteren Arbeiten parallel zur Fortführung des *Joint Research Projects "*Workload Prediction auf Mainframes" ergänzt werden. Um diese Ergänzungen zu erleichtern, ist die Bedienungsanleitung im international genormten *Open Document Format* (ODF) erstellt worden. Ein gängiges Office-Paket, welches dieses Format unterstützt und auch bei dieser Arbeite eingesetzt wurde ist das frei verfügbare *OpenOffice*.

4.2. Datenbankumstellung auf DB2 for z/OS

Die Umstellung der Datenbank von *Apache Derby* auf *DB2 for z/OS* hat sich als wichtiger Schritt erwiesen, der gleich zwei Vorteile mit sich bringt. Zum einen erfolgte der Austausch, um durch die Einführung des XML-Speicherformats die Beseitigung des *generischen Tabellenformats* vorzubereiten. Zum anderen war dies ein erster Schritt zur Migration des *WLP-Frameworks* auf den Großrechner. Die Wahl fiel deswegen bewusst auf die z/OS-Version und nicht auf die in Kapitel 3.2.2 ebenfalls erwähnte LUW-Version von DB2.

Das neue XML-Speicherformat, welches relationale Tabellen mit XML-Dokumenten kombiniert, ist exklusiv an die DB2-Datenbank gebunden. Dieses Speicherformat wurde erst durch die Einführung der *pureXML*-Technologie in DB2 Version 9.1 ermöglicht. Als weiterer Vorteil wurde durch diese neue Technologie die Elimination des *generischen Tabellenformats* möglich.

Den einzigen Nachteil stellten die in Kapitel 3.2.2 angesprochenen Besonderheiten von *DB2 for z/OS* dar, die durch die spezielle Architektur von *System Z* bedingt sind. Da aber in aller Regel nur Datenbank-Administratoren mit dem Aufsetzen und der Administration einer Datenbank auf einem Großrechner beauftragt werden, relativiert sich dieses Problem für den eigentlichen Benutzer.

Vor dem Hintergrund der oben aufgeführten gewichtigen Vorteile ist dieser Nachteil irrelevant und unbedeutend. Es bleibt festzuhalten, dass die im Rahmen der Arbeit festgelegten Ziele ausschließlich mit *DB2 for z/OS* zu erreichen waren.

4.3. Elimination des generischen Tabellenformats

Die Beseitigung des *generischen Tabellenformats* war zentraler Punkt dieser Arbeit. Als erstes wurde mit der Refaktorisierung des *RMFRecorders* begonnen, bei der die Umstellung des Speicherformats für die Metrik-Daten mit Hilfe der *JDOM*-Schnittstelle durchgeführt wurde. Die vorher relational gespeicherten Metrik-Daten werden nun pro Zeitstempel komplett in ein eigenes XML-Dokument gespeichert. Dieses Dokument wiederum wird in die Spalte einer relationalen Tabelle abgelegt, welche über den jeweiligen Zeitstempel als eindeutigen Schlüssel abgefragt werden kann. Laufzeiteinbußen durch die Umstellung auf das XML-Speicherformat sind nur marginal feststellbar und fallen nicht ins Gewicht. Bei entsprechender Optimierung der Abfragen können sie sogar ganz verschwinden. JDOM als DOM-orientierte Schnittstelle zur XML-Verarbeitung erwies sich als äußerst unkompliziert und einfach zu integrieren. Trotz der Einfachheit, ist die Schnittstelle auch für kompliziertere Aufgaben geeignet und folglich auch für künftige Weiterentwicklungen gerüstet. Zudem zeichnet sich JDOM durch freie Verfügbarkeit und weite Verbreitung aus.

Da die generierten Daten aus [Geb06] auch in dieser Arbeit als Referenzdaten verwendet wurden, musste ein Weg gefunden werden die vorhandenen relational gespeicherten Metrik-Daten in das XML-Format zu überführen. Als Ergebnis entstand das Konvertierungsprogramm *Converter*, mit dem die 8350 relationalen Referenzdatensätze aus Tabelle T060619A_DS5 in 835 Datensätze in Tabelle T060619A_DS5_XML konvertiert wurden.

Im Gegensatz zur Refaktorisierung des *RMFRecorder* bei dem "nur" die Konstruktion eines XML-Dokuments umgesetzt werden musste, gestaltete sich die Refaktorisierung der *WLPrediction* schwieriger. Es waren gleich mehrere Klassen und Funktionen von den Änderungen betroffen und der Aufwand war somit relativ hoch. Trotzdem hatte die Beseitigung des *generischen Tabellenformats* einige Vorteile zur Folge.

Zunächst einmal wurde dadurch die umfangreiche und komplexe Hilfskonstruktion beseitigt (vgl. 2.2), die für die Datenrekonvertierung aus dem *generischen* in das relationale Format zuständig war. Dieses Konstrukt erschwerte eine mögliche Weiterentwicklung aufgrund der Komplexität und des resultierenden Programmieraufwands in erheblichem Maße.

Die dieser Datenrekonvertierung zu Grunde liegende Datenstruktur einer assoziativen Tabelle wurde in diesem Zusammenhang durch einen einfachen zweidimensionalen Array ersetzt. Dies brachte nicht nur Vorteile in der Laufzeit, sondern auch in der Transparenz und dem Aufbau des gesamten *WLP-Frameworks*.

Das einzige während der Umsetzung aufgetretene Problem entstand durch die mangelnde Flexibilität des verwendeten Datentyps. Ein initialisierter Java-Array ist in seiner Größe nicht mehr veränderbar. Im vorliegenden Fall betraf dies das Löschen einer Tabellenspalte, die deswegen nur über den Umweg eines zweiten Arrays möglich war. Dieser Nachteil ist aber vernachlässigbar, weil er kein alltägliches Szenario darstellt und darüber hinaus ohne spürbare Laufzeiteinbußen umgangen werden konnte. Es ist allerdings nicht vollkommen ausgeschlossen, dass bei der zukünftigen Weiterentwicklung der Array-Datentyp an seine Grenzen stößt und ein besserer gefunden werden muss.

Nach der erfolgten Refaktorisierung wurde in der graphischen Darstellung ein weiterer Fehler gefunden: Sämtliche Vorhersagewerte waren nicht um die Anzahl der Vorhersageschritte zeitlich verzögert, da die Anzahl der Vorhersageschritte nicht an die Funktion, die für die graphische Darstellung zuständig ist, übergeben wurde. Letztendlich konnte aber auch dieses Problem behoben werden.

4.4. Migration auf den Großrechner

Die Durchführung der Migration stellte keinen großen Aufwand dar. Alle benötigten Dateien wurden automatisiert per *Ant*-Skript in das vorgefertigte dist-Verzeichnis kopiert. Dieses Verzeichnis musste im Anschluss auf das USS-Subsystem übertragen und minimal konfiguriert werden.

Durch die Migration des *WLP-Frameworks* auf den Großrechner ist auch die geographische Nähe zum WLM und zur CP, mit denen es idealerweise eng zusammenarbeiten soll, hergestellt worden. Ein Vorteil dieser Migration waren die resultierenden kürzeren Wege für das Transportieren der Daten und die Loslösung vom Client-Rechner. Darüber hinaus ermöglichte dieser Schritt eine zukünftige engere Verzahnung der drei oben aufgeführten Anwendungen, die zu neuen interessanten Möglichkeiten mit weiteren Vorteilen führen könnte.

Als nachteilig erwies sich jedoch die fehleranfällige Übertragung auf den Großrechner. Es gab einige mögliche Fehlerquellen für in diesem Gebiet unerfahrene Benutzer, die als Konsequenz in der Anleitung festgehalten wurden sowie ein paar offene Fragen. So ist z. B. das Problem mit der Wahl des Übertragungsmodus für die Datei cimmetrics.xml noch nicht abschließend gelöst worden und sollte näher untersucht werden. Zudem muss darauf hingewiesen werden, dass trotz der erfolgten Migration nicht alle zum Betrieb erforderlichen Komponenten auf dem Großrechner liefen. Der für das Erstellen und Trainieren von Agenten zuständige *ABLE-Editor* war durch die Bedienung über eine graphische Benutzeroberfläche auf dem USS-Subsystem nicht lauffähig. Als Folge musste diese Aufgabe auch weiterhin auf dem Windows-Client erledigt werden, wobei diese Tatsache kaum ins Gewicht fällt, da das Erstellen, Trainieren und Speichern eines Agenten in der Regel einmalig durchgeführt wird. Zusätzlich könnte das rechenintensive Trainieren eines Agenten auf dem Großrechner andere laufende Prozesse negativ beeinflußen. Schließlich lieferte die Laufzeit der *WLPrediction*-Anwendung auf den ersten Blick nur unbefriedigende Ergebnisse ab. Durch die oben erwähnten kürzeren Wege hätte man in der Folge eine kürzere Laufzeit erwartet, das Resultat war jedoch trotz identischem Quellcode eine eher längere Laufzeit. Dieses untypische Verhalten liegt möglicherweise zum einen an der WLM-Konfiguration und der dadurch bereitgestellten Betriebsmittel und zum anderen an der vorliegenden Systemkonfiguration. Das verwendete Testsystem *BOEWLM4* war ein per *Vicom* virtualisiertes System, das in erster Linie zum Testen von entwickelten Anwendungen dient und somit bezüglich Laufzeiten nicht repräsentativ war. Zu diesem Verhalten sollten in folgenden Arbeiten weitere Untersuchungen durchgeführt werden.

Trotz dieser bisher ungelösten Probleme überwiegen dennoch die Vorteile. Durch die erfolgreiche Migration des *WLP-Frameworks* auf den Großrechner kann zukünftig eine weitere Optimierung durch die Integration in das z/OS-Gesamtsystem mit WLM und CP erfolgen. Dies ist Voraussetzung für eine sinnvolle Fortführung des *Joint Research Projects "*Workload Prediction auf Mainframes".

5. Zusammenfassung und Ausblick

Die Fortsetzung des *Joint Research Projects* "Workload Prediction auf Mainframes" hat entscheidende Fortschritte erbracht und zu neuen richtungsweisenden Erkenntnissen geführt. Das *WLP-Framework*, welches in der Arbeit von Hagmann [Hag07] erstellt wurde und auf den theoretischen Grundlagen von Gebhard [Geb06] und Kleeberg [Kle06] basiert, konnte in wesentlichem Umfang weiterentwickelt und verbessert werden.

Durch die Erstellung einer Bedienungsanleitung wurde die Bedienung des WLP-Frameworks vereinfacht. Eine weitere Vereinfachung auf struktureller Ebene erfolgte durch die Umstellung auf die Datenbank DB2 for z/OS 9.1 mit deren neu eingeführter pureXML-Technologie. In der Folge wurde das generische Tabellenformat und alle damit verbundenen Nachteile eliminiert und durch eine Kombination aus relationalem Speicherformat und XML-Speicherformat ersetzt. Im Zuge der Refaktorisierung der beiden Anwendungen RMFRecorder und WLPrediction wurde der verwendete Datentyp einer assoziativen Tabelle durch den Datentyp eines einfachen zweidimensionalen Arrays abgelöst. Auch das Hilfskonstrukt zur Datenrekonvertierung, welches durch das Einführen des generischen Tabellenformats notwendig geworden war, konnte entfernt werden. Nachdem die Refaktorisierung abgeschlossen war, wurde mit der Migration des WLP-Frameworks auf den Großrechner begonnen. Durch die Datenbankumstellung lief die Datenbank als erste der drei Komponenten schon auf dem Großrechner. Mit der Übertragung der beiden anderen Komponenten auf die UNIX System Services (USS) wurde die Migration vervollständigt. Die Vorgehensweise dieser eigentlich trivialen Aufgabe und die dabei aufgetretenen Probleme wurden in der Bedienungsanleitung detailliert dokumentiert.

Diese Arbeit untermauert die schon aus den vergangenen Arbeiten hervorgegangenen Argumente zur Umsetzung einer Lastvorhersage auf dem Großrechner. Sie hat aber auch neue Fragen und Problemstellungen aufgeworfen und interessante Erkenntnisse geliefert. Das mittelfristige Ziel sollte eine Integration des Prototypen in die Umgebung des *Workload Managers*(WLM) und des *Capacity Provisionings* (CP) sein, da beide auf die berechneten Vorhersagewerte mit eventuell nötigen Betriebsmitteln reagieren können. Durch die Migration auf den Großrechner hat das Projekt eine wichtige Entwicklungsstufe auf dem Weg zur Produktreife erreicht und rückt diesem Ziel noch ein Stück näher. Gleichzeitig entstehen neue Aufgabenstellungen, wie z. B. eine engere Verknüpfung zwischen WLM, *WLP-Framework* und CP konkret aussehen könnte. So ist zwar eine Schnittstelle zwischen WLM und *WLP-Framework* implementiert, aber eine entsprechende Schnittstelle, um dem CP die vorhergesagten Daten zur Verfügung zu stellen, ist bislang nicht vorhanden. Da das CP ohnehin das *Common Information Model* (CIM) benutzt, um RMF-Daten abzufragen, könnte eine sinnvolle Lösung in Form eines CIM-Providers als Ausgabe-Schnittstelle in die *WLPrediction*-Anwendung eingebaut werden.

Weiterhin muss erörtert werden, ob und wie das Erstellen und Trainieren von Agenten auf dem Großrechner ermöglicht werden kann ohne dadurch andere Prozesse einzuschränken oder zu behindern. Dabei kann man zum einen versuchen den aktuell dafür eingesetzten *ABLE-Editor* so anzupassen und zu erweitern, dass er auf dem Großrechner lauffähig ist. Zum anderen kann der eingebetteten Webserver *Jetty* so erweitert werden, dass er diese Funktionalität über den Browser zur Verfügung stellt. Da ein Browser heutzutage auf jeglicher Art Rechner verfügbar ist, sollte generell die Möglichkeit einer erweiterten Nutzung von *Jetty* als eine Art Kontrollzentrale in Betracht gezogen werden. Eine zentralisierte Oberfläche im Browser, in der man von der Konfiguration des kompletten *WLP-Frameworks* über das Starten der einzelnen Anwendungen bis hin zur graphischen Darstellung der Ergebnisse alles erledigen kann, würde immense Vorteile in der Benutzung und Weiterentwicklung des *WLP-Frameworks* mit sich bringen.

Darüber hinaus sind noch nicht alle theoretischen Ergebnisse aus [Geb06] und [Kle06] in das *WLP-Framework* implementiert worden. So verwendet das eingesetzte *Feed-Forward* Netz im Trainingsalgorithmus nicht das *Konjugierte Gradienten-Verfahren mit Polak-Ribiere Updates*, sondern den *Backpropagation*-Algorithmus, und auch das zweite untersuchte neuronale Netz, das *FlexNet*, wurde noch nicht als ABLE-Bean implementiert.

Ein weiteres interessantes Gebiet betrifft die mögliche Validierung der Vorhersage-Ergebnisse, durch die automatisch Funktionen ausgelöst werden können. Die Möglichkeiten reichen, abhängig von der Qualität der berechneten Ergebnisse, von Nachtrainieren des Agenten über Ändern der verwendeten Metriken bis hin zum Wechsel des benutzten neuronalen Netzes. Dabei kann auch eine automatische Merkmals- und Modellselektion nützlich sein, die je nach vorliegendem Fall das passende Netz und die passenden Metriken auswählt. Beides ist noch nicht implementiert, kann aber eine große Wertsteigerung mit sich bringen.

Abschließend muss als einer der nächsten Schritte das *WLP-Framework* beim Kunden in einer z/OS-Installation unter realen Bedingungen getestet werden, um die Alltagstauglichkeit unter Beweis zu stellen.

A. Bedienungsanleitung für das WLP-Framework

Bedienungsanleitung für das WLP-Framework

Diese Bedienungsanleitung soll den Betrieb des WLP-Frameworks erleichtern. Es werden keine Hinweise zur Bedienung eines Großrechners gemacht, dies wird explizit vorausgesetzt! Das selbe gilt für die Bedienung von Eclipse als IDE, die hier vorgestellten SQL-Clients und die Bedienung eines FTP-Clients.

Da das verwendete System mit Vicom virtualisiert wurde und dies nicht zur allgemeinen Wissensbasis über Großrechner zählt, wird zu Beginn auch der IPL eines Vicom-Systems ausführlich beschrieben.

Diese Bedienungsanleitung ist "**work-in-progress**" und entstand während der Nutzung des WLP-Frameworks im Rahmen der Diplomarbeit "Workload Prediction für den WLM" von Semmo Bakircioglu. Sie wurde nach bestem Wissen und Gewissen angefertigt, erhebt aber dennoch keinen Anspruch auf Vollständigkeit und Korrektheit!

Inhaltsverzeichnis

1 Voraussetzungen für das Starten der WLP-Anwendungen	2
1.1 Initial Program Load (IPL)	2
1.2 RMF-Monitor, DD-Server und CIM-Server starten	10
1.3 Starten der DB2-Datenbank	18
1.4 WLPDefault.properties	19
1.5 BSO Authentication.	20
2 Herstellen einer Datenbankverbindung	22
2.1 IBM Data Studio Developer	22
2.1.1 Einrichten der Verbindung	23
2.1.2 Herstellen der Verbindung.	25
2.2 SQuirreL SQL-Client	26
2.2.1 Hinzufügen und Laden des JDBC-Treibers für DB2 for z/OS	27
2.2.2 Einrichten der Verbindung	30
2.2.3 Herstellen der Verbindung und kurze Einführung in die Benutzung von SQuirreL	33
3 Benutzung des ABLE-Editors (Projekt Ablegui)	37
3.1 WlpAgent erstellen	37
3.2 WlpAgent trainieren	<u>41</u>
3.3 WlpAgent speichern	43
4 Starten des RMFRecorders	44
5 Starten der WLPrediction	45
6 Migration auf den Großrechner	46
6.1 Vorgehensweise	46
6.2 Beenden der WLP-Anwendungen auf dem Großrechner (SIGINT)	47
7 Sonstige wichtige Hinweise und Tips	50

Seite 1 von 50

1 Voraussetzungen für das Starten der WLP-Anwendungen

Diese Voraussetzungen gelten für den Betrieb der beiden Anwendungen **RMFRecorder** und **WLPrediction**, wobei Kapitel 1.2 für *WLPrediction* keine Rolle spielt.

1.1 Initial Program Load (IPL)

Als erstes wird ein Host Account benötigt. Als Betriebssystem sollte mindestens z/OS 1.9 hochgefahren werden. Und selbstverständlich werden für die meisten Aufgaben Admin-Rechte benötigt.

1. Als erstes muss eine **tn3270 Host Session** gestartet und im geöffneten Fenster der Befehl '**d-ips'** eingegeben werden, um die verfügbaren Systeme auf Deutschland einzuschränken.

Bession A - [24	x 80]									X
File Edit View Communication Actions Window ZipPrint Help										
IBM inte	rnal onl	y l	ЕМЕАУЯ	AMP			FERMINAL=	FU0U1689	/	
		Interi	national A	oplicati	ons		VAMP Pag	e 1		
		13	:23, MONDA	Y, J	ianuary		12, 2009			
APPLNAME	STATUS	C	APPLNAME	STATUS		С	APPLNAME	STATUS		С
RHL	UNLINE	01:01	CLHIM	UNLINE	01:01		NVE32	UNLINE	01:01	
	UNLINE	01:01	СРМН	UNLINE	03:09		DISU		01:01	
SF2	UNLINE	01:01	WWCPMA	UNLINE	01:01		EITIRC	UNLINE	01:01	
BDC	ONLINE	01:01	DK-SMS	ONLINE	01:01		FTSO		01:01	
IBMNET	ONLINE	01:01	ATSO	ONLINE	04:21		HIO		01:01	
ENGINE	ONLINE	01:01	BTSO		01:01		IAS	ONLINE	01:01	
SERVICE	ONLINE	01:01	CICSKMB	ONLINE	01:01		IBMNET2	ONLINE	01:01	
CCDN	ONLINE	01:01	CICSKMF	ONLINE	01:01		IIN		01:01	
ELINK	ONLINE	01:01	CPPSIMS	ONLINE	01:01		IPO		01:01	
EHONE	ONLINE	01:01	CTSO		01:01		ISA		01:01	
EHONEGB	ONLINE	01:01	D-IPS	ONLINE	01:01		ISB		01:01	
EMEAVM1	ONLINE	01:01	DECEVAMP	ONLINE	01:01		ISC		01:01	
EMEAVM2	ONLINE	01:01	NVE	ONLINE	01:01		IS1		01:01	
EMEAVM3	ONLINE	01:01	NVE31	ONLINE	01:01		IS2		01:01	
DE4 4				DE0	D	NO				
PFI int	ernation	at HPPLS,	PF2 NURDI	C, PF3	Region	NU	(IH, PF4 K	egion so	uth	
PF5 Reg	ION WEST	, PF6 Cen	tral Regio	n	PF7,8	(SC	croll),	? or	LUGUEI	
==> d-1ps										
M <u>A</u> a				Î					24/(010
	ata aanuar haat ta 3	2270 de ibm com using l	wheel EU011699 and per	+ 22					24/(<u>, 110</u>

(Note: Der Befehl wird mainframe-typisch mit der Strg-Taste abgeschickt, nicht mit Enter!)

2. Nun kann man schauen ob das gewünschte System schon online ist. Dies kann man am **Application Status** ablesen. Wenn hinter dem Systemnamen **down** steht, dann ist der Host offline, ansonsten ist er online. Zusätzlich besteht auch noch eine

Seite 2 von 50

farbliche Unterscheidung zwischen online und down.

Das hier benutzte Testsystem WLM4 findet man, indem man erst die F6-Taste drückt um auf Page 6 zu wechseln.

Falls das gewünschte System online ist, bitte mit Schritt 10 weitermachen.

Dessio	on A - [24 x 80]							
File Edit	View Communication	Actions Window ZipPrin	t Help					
	1 2 5 2	🔳 🛋 💺 🛃 🕹	5 🛃 🗎 🌒	<i></i>				
IBM	Germany		OEBL	INGE	N	TERMINAL	=FU0U193	1/
LABC	RATORY VM9	L L	ocal App	plication		VAMP Pag	je 6	
		14	:30, Mone	day ,	January	12, 2009		
Ap	plication	Status	Appl:	ication S	tatus	Appli	cation S	tatus
WLM1		19:10	I WLMF	down	18:56	 WLMT		12:29
WLM2	2 down	18:56	WLMG		07:50	WLMU	down	18:56
WLM3	}	08:36	WLMH	down	18:56	IRD1		10:20
WLM4	down	14:30	WLMI	down	18:56	IRD2		07:54
WLM5	down	15:59	WLMJ		09:33	IRD3		08:06
WLME	down	18:56	WLMK	down	18:56	IRD4		11:50
WLM7	' down	18:56	WLML	down	18:56	IRD5		11:50
WLM8	down	18:56	WLMM	down	18:56	IRD6		14:11
WLMS	down	18:56	WLMN	down	18:56	IRD7	down	18:56
WLMA	a down	18:56	WLMO	down	18:56	IRD8	down	18:56
WLME	8 down	18:56	WLMP	down	18:56			
WLMC	down	06:45	WLMQ	down	18:56			
WLMD	down	18:56	WLMR	down	18:56			
WLME	down	18:56	WLMS	down	18:56			
Inn	ut: Applic	ation Gro	up Grou	Applica	tion Ann	lication/I		MD6/TNEO
111	PF1	- PE5 PE7	PF8 (sci	coll) P	F9 (Group	s). HELP	or 1060	FF
		110, 111	,	0.07,	ter oup.		OI LOGO	
==>								
MA	а			Î				24/005
🕤 Conne	cted to remote server/ho	st tn3270.de.ibm.com using	lu/pool FU0U1931 an	nd port 23				1

 Wenn das System down ist, dann muss man es zunächst hochfahren, bei Mainframes heisst das IPL. Das hier benutzte Testsystem ist ein virtualisiertes System (z/Vicom) in welches man sich erst einloggen muss, bevor man es hochfahren kann. Mit dem Befehl 'boevma' kommen wir in die Konsole.

Seite 3 von 50
TA Service A	[24 - 90]										
File Edit View	Communication Ac	tions Window ZipPrin	t Help								
) 🛋 🐘 🛤									
IBM Ge	rmanu		OEBL	INGE	N	TERMINAL	_=FU0U193	1/			
LABORA	TORY VM9	L	ocal App	olication	5	VAMP Pac	qe 6				
	14:30, Monday , January 12, 2009										
Application Status Application Status Application Statu											
WLM1		19:10	WLMF	down	18:56	WLMT		12:29			
WLM2	down	18:56	WLMG		07:50	WLMU	down	18:56			
WLM3		08:36	WLMH	down	18:56	IRD1		10:20			
WLM4	down	14:30	WLMI	down	18:56	IRD2		07:54			
WLM5	down	15:59	WLMJ		09:33	IRD3		08:06			
WLM6	down	18:56	WLMK	down	18:56	IRD4		11:50			
WLM7	down	18:56	WLML	down	18:56	IRD5		11:50			
WLM8	down	18:56	WLMM	down	18:56	IRD6		14:11			
WLM9	down	18:56	WLMN	down	18:56	IRD7	down	18:56			
WLMA	down	18:56	WLMO	down	18:56	IRD8	down	18:56			
WLMB	down	18:56	WLMP	down	18:56						
WLMC	down	06:45	WLMQ	down	18:56						
WLMD	down	18:56	WLMR	down	18:56						
WLME	down	18:56	WLMS	down	18:56						
Toput	Applica	tion Gro	up Grou	Applica	tion Appl	ication/1					
Input	. пррсіса РЕ1 -	PES PE7	PF8 (scr	oll) P	F9 (Groups) HELE	2 or 1060	FF			
)110 (30	0(())	e (di odpa		or Lodo				
==> boe	vma										
MA a				Ŷ				24/011			
Connected to	remote server/host t	tn 3270. de. ibm. com using	lu/pool FU0U1931 an	d port 23				1			

4. Dort wechselt man auf die Zeile **Command** ganz unten und gibt folgenden Befehl ein: 'I Hostname by UserID here'.

In den meisten Fällen entspricht die UserID für die Konsole **NICHT** der oben erwähnten UserID, die man für das einloggen auf dem Host verwendet, da Erstere erweiterte Rechte benötigt.

Im vorliegenden Fall würde der Befehl so aussehen:

'l wlm4 by bakir (here)'

Das ist **'here'** optional, für den Fall eines **Reconnects**. Im Anschluss wird wie gewöhnlich ein Passwort abgefragt.

과 Session A - [43 x 80]	
File Edit View Communication Actions Window ZipPrint Help	
<pre>z/VM ONLINE ***********************************</pre>	
Please fill in your Userid and Password Userid ===> Password ===>	
Command ===> l wlm4 by bakir here	EVMA
MA a û g ⁽¹⁾ Connected to remote server/host th3270.de.ibm.com using lu/pool FU0U1931 and port 23	42/037

5. Unter Umständen muss nach der Passwortbestätigung wiederholt **Strg** gedrückt werden. Falls am rechten unteren Rand nach Drücken der Strg-Taste **More...** stehen sollte, dann muss zum weiterkommen die **Pause**-Taste gedrückt werden. Mit Hilfe der Kombination beider Tasten kommt man dann zum nächsten Screen

Seite 5 von 50

3 Session A - [43 x 80]	
File Edit View Communication Actions Window ZipPrint Help	
L WLM4 BY BAKIR HERE	
Enter your password,	
or To change your password, enter: ccc/nnn/nnn	
where ccc = current password, and nnn = new password ICH70001I WLM4 LAST ACCESS AT 14:29:48 ON MONDAY. JANUARY 12. 2009	
z/VM Version 5 Release 3.0, Service Level 0000 (64-bit),	
Duilt on IBM Virtualization rechnology There is no logmsg data	
FILES: NO RDR, NO PRT, NO PUN	
z/VM V5.3.0 2008-06-02 14:33	
DMSACP723I C (300) R/O	
Ready; T=0.01/0.01 14:37:52	
DMSACP7231 V (1210) R/O	
DMSACP723I W (1211)_R/0	
DMSACP723I X (45D) R/O	
DMSACF7231 T (307) R/O	
DMSACP723I Q (309) R/O	
DMSACP723I P (310) R/O	
DMSACF7231 0 (311) R/0	
DMSACP7231 M (313) R/0	
DMSAGP723I L (314) R/O	
DMSACP723I K (315) R/O	
DMSACF7231 J (317) R/O	
DMSACP060E File not found; filemode H(318) will not be accessed	
Ready; T=0.01/0.01 14:37:52	
More BOEV	/MA
Connected to remote server/host th3270.de.ibm.com using lu/pool E10011931 and port 23	427001

Im Screen Select Operating System for Test Machine muss man unter Op. Syst.
 'm' angeben, unter VSD 'n' und und abschließend unter MVS Rel. die Nummer für das gewünschte Release. Per Voraussetzung sollte mindestens z/OS 1.9 oder höher ausgewählt werden. Im vorliegenden Fall wird z/OS 1.9 mit Release '23' gewählt Bei den restlichen Einstellungen kann man die Default-Werte übernehmen und mit der Strg-Taste bestätigen.

₽ ¹ Session A - [43 x 80]	
File Edit View Communication Actions Window ZipPrint Help	
Select Uperating System for lest Ma	Chine WLM4
$\Omega_{\rm D}, {\rm Sust.} === > m$ C - CMS M - MVS	
VSD ===> n N - No Y - Yes	
MVS Rel. ===> 23 Required for MVS	
MVS 1.0.M SERVLVL 0842	Z/OS 1.10 MRES A
2 MVS 1.0.0 02 SERVLVL 0901	WLMTEST ZOS R10
3 MVS 1.0.0 SERVLVL 0901	Z/OS 1.10
4 MVS 1.1.0 07 SERVLVL 0851	R11 DR07 COMMON
5 MVS 1.1.0 07 SERVLVL 0851	R11 DR07 RELPRO
	RIZ DROI RELPRO
	7/08 1 4
	LIMTEST R141
10 MVS 1.5 M SERVIVI 0832	7/08 1.5
11 MVS 1.6.M SERVLVL 0820	Z/08 1.6
12 MVS 1.6.1 SERVLVL 0901	WLMTEST R161
13 MVS 1.7.M SERVLVL 0720	Z/0S 1.7 MRES A
14 MVS 1.7.0 SERVLVL 0836	R17 JAPANISCH HIR
15 MVS 1.7.0 SERVLVL 0901	WLMTEST R170
16 MVS 1.7.0 SERVLVL 0901	Z/OS 1.7
17 MVS 1.7.1 SERVLVL 0901	WLMTEST R171
18 MVS 1.8.0 SERVLVL 0901	WLMTEST R180
19 MVS 1.8.0 SERVLVL 0901	2708 1.8
20 MVS 1.9.M SERVLVL 0833	2708 1.9 MRES B
21 MVS 1.9.0 SERVLVL 0839	Z7US 1.9 WLM GPU
23 MVS 1 9 6 SERVIVI 6961	7/08 1 9
	US(398 Z.8
25 MVS 6.9.M SERVIVI 0901	08/390 2.9
26 MVS 7.1.M SERVLVL 0901	08/390 2.10
27 MVS 8.1.M SERVLVL 0839	Z/0S 1.1
28 MVS 8.2.M SERVLVL 0901	Z/0S 1.2
29 MVS 1.4.0 SERVLVL 0606	Z/0S 1.4 FMS1
30 MVS 1.4.0 SERVLVL 0705	Z/OS 1.4 WBC1
31 MVS 1.6.0 SERVLVL 0635	Z/OS 1.6 WBB1
32 MVS 1.7.0 SERVLVL 0720	Z/OS 1.7 WBD1
SYSPARM ===> N N Noprompt Y Prompt	
Indice EES No. UX IODE Noprompt IX IODE prompt	
Storage 1024 Megabytes Storage max 1024	
CPs ==> 8 Central Processor Unite max 8	
Console ===> N Y separate address N same addr	ess
	28/016
	307016
Connected to remote server/host tn32/0.de.ibm.com using lu/pool FU0U1931 and port 23	

(**Note:** Im Zuge der Umstellung von **Apache Derby** auf **DB2**, sollte der Wert für Storage per Default mindestens auf **1024 MB** stehen!)

7. Nun kommt man durch wiederholtes Drücken der **Strg-** und **Pause-**Tasten zum unten angezeigten **ipl MVS** Screen. Nach kurzer Zeit sollte unten rechts statt **Running** wieder **More...** stehen. Durch erneutes Drücken von **Strg** startet dann der eigentliche IPL, der einige Minuten in Anspruch nehmen kann.

Seite 7 von 50

30 Session A - [43 x 80]	_ 🗆 🖂
File Edit View Communication Actions Window ZipPrint Help	
ipl MVS 1.9.0 from 5028 CL LOADPARM 410000M1	
Ready; T=0.29/0.31 14:48:16 00: DASD 0190 DETACHED	
00: DASD 0191 DETACHED	
00: DASD 019E DETACHED	
00: STORAGE = 16 00: Storage cleared - system reset.	
00: CONS 0009 DETACHED	
00. CONS 0100 DEFINED	
More BOEV	MA
GP Connected to remote server/host tn3270.de.ibm.com using lu/pool FU0U1931 and port 23	427001

- 8. Wenn man sicher gehen will, dass der IPL nun abgeschlossen ist, kann ein weiteres Terminal-Fenster geöffnet werden und durch wiederholen der Schritte 1-3 der Status des Systems wiederholt werden.
- Bevor nun die Konsole geschlossen wird, sollte eine Abmeldung erfolgen. Das wird durch Drücken der Escape-Taste gefolgt von der Eingabe des Befehls 'disconnect' und abschließender Bestätigung durch drücken der Strg-Taste erledigt. Die Konsole kann nun geschlossen werden.

Seite 8 von 50

₽ <mark>₽</mark> Session A - [43 x 80]												
File Edit View Communication Actions Window ZipPrint	Help											
	a 🗎 🖉 🏈	2										
01: HCPGSP2627I The virtual	machine	is	placed	in	CP	mode	due	to	а	SIGP	initia	ιc
PU reset from CPU 00. 02. HCPGSP2627I The virtual	machine	ie	placed	in	CP	mode	due	to	-	STGP	initia	1 C
PU reset from CPU 00.	macritic	1.0	proced		01	mouc	auc		u	0101	101010	
03: HCPGSP2627I The virtual	machine	is	placed	in	CP	mode	due	to		SIGP	initia	ιc
PU reset from CPU 00.	machina	10	placed	in	CP	mode	dua	+ ~	-	STGD	initia	1 0
PU reset from CPU 01.	machine	13	praced	111	0F	mode	due	.0	a	GIUF	1111(18	
05: HCPGSP2627I The virtual	machine	is	placed	in	CP	mode	due	to		SIGP	initia	ιc
PU reset from CPU 03.					CD					er en		1 0
PU reset from CPU 03.	machine	15	placed	IN	GP	mode	aue	to	а	SIGP	initia	ιu
07: HCPGSP2627I The virtual	machine	is	placed	in	CP	mode	due	to		SIGP	initia	ιc
PU reset from CPU 04.	the second second			100	on	2822-261				01.00		
CPU 00.	machine	15	placed	ın	GР	mode	aue	το	а	SIGP	зтор т	rom
00: HCPGSP2627I The virtual	machine	is	placed	in	CP	mode	due	to		SIGP	initia	ιс
PU reset from CPU 04.					-							
DU: HCPGSP26271 The virtual BU reset from CBU 03	machine	15	placed	ın	GP	mode	due	to	а	SIGP	initia	LU
2 <u>1</u> 12												
disconnect_												
							(Sp F	Rea	id E	BOEVMA	
MA a			Û								427	011
Connected to remote server/host tn3270.de.ibm.com using lu/	pool FU0U1931 and p	ort 23										11

 Wenn das System schließlich online ist, kann der Login mit dem Befehl 'Hostname UserID' erfolgen. Für den vorliegenden Fall wäre das 'wIm4 prak1'. Nach dem Login-Befehl erfolgt die obligatorische Passwort-Abfrage.

Seite 9 von 50

B Session A -	[24 x 80]										
File Edit View	Communication Ac	tions Window ZipPrint	Help								
	Æ 🛼 🔳 🔳) 🛋 🐚 🖦 💩	' 🛃 👜 🌒	<i></i>							
IBM Ger	many	В	OEBL	INGE	N	TERMINAL	_=FU0U310	4/			
LABORAT	ORY VM9	L	ocal App	lication	5	VAMP Pag	ge 6				
15:03, Monday , January 12, 2009											
Application Status Application Status Application Status											
WLM1		19:10	WLMF	down	18:56	WLMT		12:29			
WLM2	down	18:56	WLMG		07:50	WLMU	down	18:56			
WLM3		08:36	WLMH	down	18:56	IRD1		10:20			
WLM4		14:53	WLMI	down	18:56	IRD2		07:54			
WLM5	down	15:59	WLMJ		09:33	IRD3		08:06			
WLM6	down	18:56	WLMK	down	18:56	IRD4		11:50			
WLM7	down	18:56	WLML	down	18:56	IRD5		11:50			
WLM8	down	18:56	WLMM	down	18:56	IRD6		14:11			
WLM9	down	18:56	WLMN	down	18:56	IRD7	down	18:56			
WLMA	down	18:56	WLMO	down	18:56	IRD8	down	18:56			
WLMB	down	18:56	WLMP	down	18:56						
WLMC	down	06:45	WLMQ	down	18:56						
WLMD	down	18:56	WLMR	down	18:56						
WLME	down	18:56	WLMS	down	18:56						
Input:	Applica	tion. Gro	up. Groum	Applica	tion. Appl	ication/]	NFO. VA	MP6/INFO			
anpaci	PF1 -	PF5, PF7	,PF8 (scr	roll), P	F9 (Group⊆), HELF	or LOGO	FF			
==> wlm4	prak1_										
MA a				Ŷ				24/015			
Connected to	remote server/host i	tn3270.de.ibm.com using l	u/pool FU0U3104 an	d port 23				1			

1.2 RMF-Monitor, DD-Server und CIM-Server starten

Falls nicht schon geschehen, sollten nacheinander die Komponenten **RMF**, **DDS** und **CIM** gestartet werden:

11. Es hat sich als überaus nützlich erwiesen, Befehle direkt im SDSF Log auszuführen, denn es kann in der Folge das Resultat direkt überprüft werden. Um in das **SDSF** zu kommen muss man im System Master Application Menu **'sd'** eingeben und im folgenden Untermenü **'log'** oder direkt im ersten Screen **'sd;log'**.

Seite 10 von 50

3 Session A - [43 x 80]					
File Edit View Communication	Actions Window ZipPrint	Help			
	🔳 🖬 💺 👪	e 🗎 🔌 🤣			
OPTION ===>	YSTEM MASTER	APPLICATION	MENU		
OPTION ===> L LOCAL U USER P PDF SD SDSF SM SMP/E R RACF I ISMF IP IPCS PM RMF S DFSORT PP PPS H HCD E ESCM O OMVS DB DB2 MQ MQM IM IMS CA Candle X EXIT	d;log - Products, - User selec - ISBF/Progr - System Dis - SMP/E Dial - Resource A - Interactiv - Performanc - Data Facil - IBM Progra - Lang : ENG - ESCON MANA - Open Editi - DB2 - MQSeries - IMS - Candle Ome - Terminate	TOOLS and Ut tion panel am Developm play and Sec ogs ccess Contro e Storage Ma e Problem Cd e Monitor Rt ity Sort m Products Trace ==== GER DIALOG on gamon ISPF using T	tilities ent Facility arch Facility ol Facility anagement Facility ontrol Facility 4F Rel.===> 190 => N Y/N List/log default	testm : system : runs at: mtype : sysres : wvs rel: ty dfsms : vtam : applid term.ad: racf : ispf : userid : ip addr: s jdate :	WLM4 BOEWLM4 G14/LP2=VMA 2084-314 190901 Z/OS 01.09.00 JES2 Z/OS 1.9 6.1.9 IPYA4T FU0U3104 Z/OS 01.09.0 3.09.0 ISPF 5.9 PRAK1 15:19 2009/01/12 9.152.87.20 2009.012
F1=HELP F7=UP	F2=SPLIT F8=DOWN	F3=END F9=SWAP	F4=RETURN F10=LEFT F	F5=RFIND 11=RIGHT	F6=RCHANGE F12=RFTRTEVE
MA a	P D DOWN	1 5 GWHP		TT REGIN	02/020
Connected to remote server/h	iost tn3270.de.ibm.com using lu,	pool FU0U3104 and port 23			1

12. Starten des RMF Monitor (III) mit dem Befehl '/s(tart) rmf'.

Seite 11 von 50

Session A - [43	x 80]							
File Edit View Com	munication	Actions Window Zipł	Print Help					
	Sa 🔳	🔳 📾 💺 🔜	1 1 1					
Display	EI	lter View	Print Opt	ions <u>H</u> el	p			
SDSE SAST	00	252.102 .	LM4 WLM4 01	/12/2009	0W 169	COL	JMNS 1	80
COMMAND I	NPUT	===> /s rn	of			S(CROLL ===	-> CSR
1000000		00012	16-03:38.26	81C00435	00000090	\$HASP395	JESWIRDS	S ENDED
N 0000000	WLM4	09012	15:03:38.27	07000405	00000290	TENARAT S	SLIP TRAF	1D=X33
N 0200000	WLM4	09012	15:03:30.27	51600435	00000090	\$HH5P250	JESWIRDS	
NROOOOOOO		09012	15.12.23.17	INTERNAL	000000000	\$HH3P001	LUGUNI	HPPLI
NROOOOOOO		09012	15.12.23.11	INTERNAL	0000000000	BHHSP000	CINEGO	WATNIE
SP	W L 114	05012	13.12.28.10	INTERNE	000000000	RC=08	ФЗN, H-1	PHINOL
NRAAAAAAAA	LI MA	09012	15.12.23 18	TNTERNAL	00000090	SHASP670	\$SN SOCK	ET=SAP9
SR	W LIII-	03012	10.12.20.10	INTERMITE	00000000	A VALTD	TPADDR	RC=15
NC0000000	WLM4	09012	15:12:23.18	INTERNAL	00000290	D ASM		
MR0000000	WLM4	09012	15:12:23.18	INTERNAL	00000090	IEE2001 :	15.12.23	DISPLAY
LR				998	00000090	TYPE	FULL STA	T DEV
DR				998	00000090	PLPA	22%	DK 5414
DR				998	00000090	COMMON	0% 0	DK 5414
DR				998	00000090	LOCAL	0% 0	DK 5514
ER				998	00000090	PAGEDEL (COMMAND I	IS NOT A
NC0000000	WLM4	09012	15:19:16.41	INSTREAM	00000290	LOGON		
N 0200000	WLM4	09012	15:19:37.17	TSU00438	00000291	\$HASP100	PRAK1	ON TSO
N 4000000	WLM4	09012	15:19:37.21	TSU00438	00000090	\$HASP373	PRAK1	STARTE
N 0000000	WLM4	09012	15:19:37.21	TSU00438	00000090	IEF1251	PRAK1 - L	_OGGED 0
N 0000000	WLM4	09012	15:19:37.78		00000291	IEF196I	(EF237I 5	5AA1 ALL
N 0000000	WLM4	09012	15:19:37.78		00000291	IEF196I	[EF285I	WLM.BH
N 0000000	WLM4	09012	15:19:37.78		00000291	IEF196I	LEF2851	VOL SE
N 0000000	WLM4	09012	15:19:41.44		00000291	IEF1961	GD103I S	SMS ALLO
N 0000000	WLM4	09012	15:23:07.16	STC00413	00000090	IEF4041	BPXAS - E	ENDED -
N 0000000	WLM4	09012	15:23:07.16	STC00412	00000090	1EF4041 H	SPXAS - E	NDED -
N 0000000	WLM4	09012	15:23:07.16	STC00410	00000090		SPXAS - E	ENDED -
N 4000000	WLM4	09012	15:23:07.16	51000413	00000090	2HH2P395	BPXHS	ENDED
N 4000000		09012	15.23.07.16	STC00412	000000000	30000000000000000000000000000000000000	BPAHa	ENDED
N 4000000		09012	15.23.07.18	31600410	00000090	TEAGRAT (SITE TEAD	ID=V99
N 0000000	ML MA	09012	15.23.07.18		00000290	TEA9891	SLID TRAD	D TD=X33
N 0000000	WI M4	09012	15:23:07 18		00000290	IFA989I	SLIP TRAP	D=X33
N 0200000	WLM4	09012	15:23:07.19	STC00413	00000090	\$HASP250	BPXAS PL	JRGED
N 0200000	WLM4	09012	15:23:07.19	STC00410	00000090	\$HASP250	BPXAS PU	JRGED
N 0200000	WLM4	09012	15:23:07.19	STC00412	00000090	\$HASP250	BPXAS PL	JRGED
******	****	******	кжжжжжж ВОТ	TOM OF DA	ТА жжжжжжжж	******	******	*****
F1=HELP		F2=SPLIT	F3=END	F4=RI	ETURN F	5=IFIND	F6=B00)K
F7=UP		F8=DOWN	F9=SWAP	F10=L	EFT F1:	1=RIGHT	F12=RET	RIEVE
MA a				Û				04/027
Connected to remo	te server/ho	ost tn3270.de.ibm.com us	ing lu/pool FU0U3104 and no	rt 23				4
			a speak and po					10

13. Nach der Ausführung des Befehls sollte eine **Command Issued** Meldung und als **Response** eine **RMF: Active** Meldung erscheinen.

3 Session A - [43 x 80]						🛛
File Edit View Communication	Actions Window ZipPrint Help					
<u>D</u> isplay <u>F</u> i	lter <u>V</u> iew <u>P</u> rin	t <u>O</u> ption	ns <u>H</u> elp	>		
				460		
COMMOND INDUIT	352.102 WLM4 W	LM4 0171.	272009 0	10 I D 3	COMMAND ISSUED	N COD
RESPONSE = WI MA			F		3CROLE	CON
N 0000000 WIM4	04012 15:03	:38.27		00000290	TEA989T SLIP TRAP	TD=X33
N 0200000 WLM4	09012 15:03	:38.27 8	TC00435	00000090	SHASP250 JESWIRDS	PURGED
NR0000000 WLM4	09012 15:12	:23.17 I	NTERNAL	00000090	SHASP881 LOGON1	APPLI
NR0000000 WLM4	09012 15:12	:23.17 I	NTERNAL	00000090	SHASP880 LINE80	UNIT
NR0000000 WLM4	09012 15:12	:23.18 I	NTERNAL	00000090	\$HASP679 \$SN, A=IP	VATNJE
SR					RC=08	
NR0000000 WLM4	09012 15:12	:23.18 I	NTERNAL	00000090	\$HASP670 \$SN, SOCK	ET=SAP9
SR					A VALID IPADDR,	RC=15
NC0000000 WLM4	09012 15:12	:23.18 I	NTERNAL	00000290	D ASM	
MR0000000 WLM4	09012 15:12	:23.18 I	NTERNAL	00000090	IEE200I 15.12.23	DISPLAY
LR			998	00000090	TYPE FULL STA	T DEV
DR			998	00000090	PLPA 22% 0	K 5414
DR			998	00000090	COMMON 0% O	K 5414
DR			998	00000090	LOCAL 0% O	K 5514
ER			998	00000090	PAGEDEL COMMAND I	S NOT A
NC0000000 WLM4	09012 15:19	:16.41 I	NSTREAM	00000290	LOGON	
N 0200000 WLM4	09012 15:19	:37.17 T	SU00438	00000291	SHASP100 PRAK1	ON TSO
N 4000000 WLM4	09012 15:19	:37.21 T	SU00438	000000000	\$HASP373 PRAK1	STARTE
N 0000000 WLM4	09012 15:19	:37.21 T	SU00438	00000090	IEF125I PRAK1 - L	OGGED O
N 0000000 WLM4	09012 15:19	:37.78		00000291	IEF196I IEF237I 5	AA1 ALL
N 0000000 WLM4	09012 15:19	:37.78		00000291	IEF1961 IEF2851	WLM.BH
N 0000000 WLM4	09012 15:19	:37.78		00000291	IEF1961 IEF2851	VUL SE
N 0000000 WLM4	09012 15:19	:41.44	T 0 0 0 4 4 0	00000291	TEF1961 IGD1031 S	MS ALLU
N 0000000 WLM4	09012 15:23	07.16 5	TC00413	00000090	IEF4041 BPXHS - E	NDED -
N 0000000 WLM4	09012 15:23	07.16 5	1000412	00000090	IEF4041 BPAHS - E	NDED -
N 4000000 WLM4	00012 15.23	07 16 8	TC00410	000000000	CHACHOOL DRANG - E	ENDED
N 4000000 WLM4	09012 15.23	.07 16 8	TC00413	000000000		ENDED
N 4000000 WLM4	09012 15.23	07 16 8	TC00412	0000000000	\$HASD395 BDXAS	ENDED
N AAAAAAA WLM4	09012 15:23	·07 18	1000410	0000000000	IFA989I SLIP TRAD	TD=X33
N 0000000 WLM4	09012 15:23	:07.18		00000290	IFA989I SLIP TRAP	TD=X33
N 0000000 WIM4	09012 15:23	:07.18		00000290	IEA989I SLIP TRAP	ID=X33
N 0200000 WLM4	09012 15:23	:07.19 S	TC00413	00000090	SHASP250 BPXAS PU	RGED
N 0200000 WLM4	09012 15:23	:07.19 S	TC00410	00000090	SHASP250 BPXAS PU	RGED
N 0200000 WLM4	09012 15:23	:07.19 S	TC00412	00000090	SHASP250 BPXAS PU	RGED
***********	************ <u>***</u> *	** BOTTO	M OF DAT	А жжжжжжжж	***************	*****
F1=HELP	F2=SPLIT F3	=END	F4=RE	TURN F5	5=IFIND F6=B00	К
F7=UP	F8=DOWN F9	=SWAP	F10=LE	EFT F11	=RIGHT F12=RET	RIEVE
MA			Û			04/021
Connected to remote server h	ost tn3270.de.ibm.com.using.lu/nool El	10U3104 and port 23	1			1
in the second se						

14. Als nächstes sollte der **RMF Distributed Data Server (GPMSERVE)** gestartet werden. Dies wird mit Hilfe des Befehls **'/s gpmserve'** erledigt.

Seite 13 von 50

-					
Session A - [43 x 80]					
File Edit View Communication	Actions Window Zip	Print Help			
	🔳 🛋 💺 🎭	1 1 1			
Display Fil	ter View	Print Opt:	ions <u>H</u> el	р	
SDSF SYSLOG	352 102 1	<u>11 MA 111 MA</u> 01,	12/2009	0W 169	COMMAND ISSUED
COMMAND INPUT	===> /s g	omserve	rur-		SCROLL ===> CSR
N 0000000 LIMA	A A A A A A A A A A A A A A A A A A A	15.02.28 27	LVE	00000290	TEAGET SITE TEAD TO - YO
N 0200000 WLH4	09012	15.03.30.27	STC00435	00000290	SHASP250 JESWIRDS DURGE
NRAAAAAAA WLM4	09012	15.12.23 17	INTERNAL	000000000	SHASP881 LOGON1 APPL
NR0000000 WI M4	09012	15:12:23.17	INTERNAL	00000090	SHASP880 LINE80 UNI
NR0000000 WLM4	09012	15:12:23.18	INTERNAL	00000090	\$HASP679 \$SN.A=IPVATNJE
SR					RC=08
NR0000000 WLM4	09012	15:12:23.18	INTERNAL	00000090	\$HASP670 \$SN, SOCKET=SAP
SR					A VALID IPADDR, RC=15
NC0000000 WLM4	09012	15:12:23.18	INTERNAL	00000290	D ASM
MR0000000 WLM4	09012	15:12:23.18	INTERNAL	00000090	IEE200I 15.12.23 DISPLA
LR			998	00000090	TYPE FULL STAT DE
DR			998	00000090	PLPA 22% OK 541
DR			998	00000090	COMMON 0% OK 541
DR			998	00000090	LOCAL 0% OK 551
ER			998	00000090	PAGEDEL COMMAND IS NOT
NC0000000 WLM4	09012	15:19:16.41	INSTREAM	00000290	LOGON
N 0200000 WLM4	09012	15:19:37.17	15000438	00000291	SHASP100 PRAK1 ON IS
N 4000000 WLM4	09012	15:19:37.21	15000438	00000090	SHASP373 PRAKI STARI
N 0000000 WLM4	09012	15:19:37.21	15000436	00000090	TEFIZSI PRHKI - LUGGED
N 0000000 WLM4	09012	15:19:37.70		00000291	TEF1961 TEF2371 SHHI HL
N 0000000 WLM4	09012	15:19:37.78		00000291	TEF1961 TEF2651 WLM.E
N 0000000 WLM4	09012	15.19.01.44		00000291	
	09012	15.23.07 16	STC00413	000000201	TEFANAT RDYAS - ENDED -
N AAAAAAA WLM4	09012	15.23.07 16	STC00412	0000000000	IEF404I BPXAS - ENDED -
N AAAAAAA WI M4	09012	15:23:07 16	STC00410	000000000	IEF4041 BPXAS - ENDED -
N 4000000 WIM4	09012	15:23:07 16	STC00413	00000090	SHASP395 BPXAS ENDED
N 4000000 WLM4	09012	15:23:07.16	STC00412	00000090	SHASP395 BPXAS ENDED
N 4000000 WLM4	09012	15:23:07.16	STC00410	00000090	SHASP395 BPXAS ENDED
N 0000000 WLM4	09012	15:23:07.18		00000290	IEA989I SLIP TRAP ID=X3
N 0000000 WLM4	09012	15:23:07.18		00000290	IEA989I SLIP TRAP ID=X3
N 0000000 WLM4	09012	15:23:07.18		00000290	IEA989I SLIP TRAP ID=X3
N 0200000 WLM4	09012	15:23:07.19	STC00413	00000090	\$HASP250 BPXAS PURGED -
N 0200000 WLM4	09012	15:23:07.19	STC00410	00000090	\$HASP250 BPXAS PURGED -
N 0200000 WLM4	09012	15:23:07.19	STC00412	00000090	\$HASP250 BPXAS PURGED -
N 0000000 WLM4	09012	15:28:13.34	TSU00438	00000290	IEA630I OPERATOR PRAK1
F1=HELP	F2=SPLIT	F3=END	F4=R	ETURN F	F6=BOOK
F7=UP	F8=DOMN	F9=SWAP	F10=L	EFT F1:	I=RIGHT F12=RETRIEVE
MA a			Û		04/03
Connected to remote server/ho	ost tn3270.de.ibm.com u	sing lu/pool FU0U3104 and por	t 23		

15. Es passier offenbar nichts und es erscheint die Meldung **No Response Received**. Das ist normal und sollte nicht beunruhigen.

Seite 14 von 50

Ð	Session A	- [43 x 80]																-	
File	Edit View	Communic	ation A	Actions	Window	ZipPr	rint H	elp												
6			en le		1			1 163	1 38											
				■	1 20	10	00 0	5		1										
	Disp	lay	<u>F</u> il [.]	ter	Vie	ew	Pr	int	0	pti	ions	He	lp							
	Der e	VELOG		252	1.01	о н	1 14	1.11	MA	01	197	2000	01.1		1 6 9		IONSE	DECI	TVI	-D
1 6	OMMAN		ит з	===>	. 102	2 W	LI14	WL	r14	017	121	2005	OW		105	NO RESP	UNAL	REGI	- 1 VI	
Ē	ESPON	SF=WI	M4		FRE	B10	ΩT	RMF	: A	стт	VE						ONOLL			JOIN
N	00000	00 WL	M4		0901	12	15:	03:	38	27			00	0002	90	IEA989I	SLIP	TRA)=X33
N	02000	00 WL	M4		0901	12	15:	03:	38.	27	STC	0043	5 00	00000	90	\$HASP250	JESW	TRDS	S PU	JRGED
NF	00000	00 WL	M4		0901	12	15:	12:	23.	17	INT	ERNA	L 00	00000	90	\$HASP881	LOGO	N1	f	APPLI
NF	00000	00 WL	M4		0901	12	15:	12:	23.	17	INT	ERNA	L 00	00000	90	\$HASP880	LINE	80		UNIT
NF	00000	00 WL	M4		0901	12	15:	12:	23.	18	INT	ERNA	L 00	00000	90	\$HASP679	\$SN,	A=IP	VA	INJE
SF																RC=08				
NF	00000	00 WL	M4		0901	12	15:	12:	23.	18	INT	ERNA	L 00	00000	90	\$HASP670) \$SN,	SOCI	KET:	SAP9
SF																A VALIE) IPAD	DR,	RC	=15
NC	00000	00 WL	M4		0901	12	15:	12:	23.	18	INT	ERNA	L 00	0002	90	D ASM				
MF	00000	00 WL	M4		0901	12	15:	12:	23.	18	INT	ERNA	L 06	00000	90	IEE200I	15.12	.23	DIS	SPLAY
LF												99	8 06	00000	90	TYPE	FULL	STA	τŕ	DEV
DF												99	B 00	00000	90	PLPA	22%	(рĸ	5414
DF												99	B 00	00000	90	COMMON	0%	(эк	5414
DF												99	B 06	00000	90	LOCAL	0%	(эк	5514
EF												99	8 06	00000	90	PAGEDEL	COMMA	ND :		NOT A
INC	00000	00 WL	M4		0901	12	15:	19:	16.	41	INS	TREA	MOG	00002	90	LOGON	in and the second			an asalas
N	02000	00 WL	M4		0901	12	15:	19:	37.	17	180	0043	8 06	00002	91	SHASP100	PRAK	1	0	1 1 2 0
N	40000	00 WL	M4		0901	12	15:	19:	37.	21	180	0043	8 06	00000	90	\$HASP373	PRAK	1	8	IARIE
N	00000	00 WL	M4		0901	12	15:	19:	37.3	21	180	0043	8 00	00000	90	1EF1251	PRAK1		OGG	JED O
N	00000	UU WL	M4		0901	12	15:	19:	37.	10			00	00002	91	1EF1961	IEF23	$\frac{1}{2}$	HH:	LALL
N	00000	OO WL	M4		0901	12	15:	19:	37.	10			00	00002	91	1EF1961	IEF28	51	W	LM BH
N	00000		M4		090	12	10:	19:	31.	10			00	00002	91	1EF1961	IEF20	1 C	NO	JL SE
	00000		M4		0 0 0 1	1 2	10.	17.	41.	44	et c	0041		00002	21		DDVAG	31 3		HLLU
	00000		M4		0 0 0 0	10	10.	20.	07.	10	OTC	0041		00000	90		BRAHS			
N	00000		M4		090	1 2	15.	20.	07.	16	erc	0041		00000	90	1EF4041	BBYAS	_ [-D -
N	40000		MA		0901	19	15.	20.	07	16	STC	0041	3 00	00000	90	\$H08039F	BBYA	e '		NDED
N	40000		MA		0901	12	15.	23.	07	16	STC	0041	2 00	00000	9 A	SHASP395	BPXA	202	F	
N	40000	00 11	MA		0901	12	15.	23	07	16	STC	0041		0000	9 Å	SHASP395	BPXA	s	F	NDED
N	00000	00 41	MA		0901	12	15.	23	07	18		0041	0.0	10002	9 Å	TEA989T	SLIP	TRAP		1=X33
N	00000	00 WI	M4		0901	12	15:	23:	07	18			0.0	00002	90	IEA9891	SLIP	TRA)=X33
N	00000	00 WI	M4		0901	12	15:	23:	07.	18			0.0	00002	90	IEA9891	SLIP	TRA	D II)=X33
N	02000	00 WL	M4		0901	12	15:	23:	07.	19	STC	0041	3 00	00000	90	\$HASP250	BPXA	S PU	JRG	ED
N	02000	00 WL	M4		0901	12	15:	23:	07.	19	STC	0041	0 00	0000	90	\$HASP250	BPXA	S PU	JRG	ED
N	02000	00 WL	M4		0901	12	15:	23:	07.	19	STC	0041	2 00	00000	90	\$HASP250	BPXA	S PU	JRG	ED
N	00000	00 WL	M4		0901	12	15:	28:	13.	34	TSU	0043	8 00	0002	90	IEA6301	OPER	ATO	R PI	RAK1
	F1=HE	LP	F	F2 = S	PLIT	Т		F3=	END			F4=1	RETU	JRN	F	5=IFIND	F6	= 800	ЭK	
	F7=UP		F	F8=D	IOWN			F9=	SWA	р		F10 =	LEFT		F1:	1=RIGHT	F12	=RE	RI	EVE
MA	a											Û							0	4/021
-0	Connected t	o remote ser	ver/host	t tn3270.	de ibm.o	om usir	na lu/na	ol FUOL	J3104 ar	nd por	t 23									4
1000							, p.c												_	14

(**Zu beachten:** Die UserID, mit der man den Server startet benötigt zwingend ein **OMVS Segment** in RACF und ein **Home Directory** im USS Filesystem. Eventuelle Fehlermeldungen bitte darauf überprüfen!)

16. Schließlich muss als Letztes der **Pegasus CIM-Server** mit dem Befehl **'/s cfzcim'** gestartet werden.

Seite 15 von 50

⇒ Session A - [43 x 80]			
File Edit View Communication Actions Wind	low ZipPrint Help		
	b 😓 💩 🛃 🗎 🌒 🏈		
<u>D</u> isplay <u>F</u> ilter <u>V</u>	iew <u>P</u> rint <u>O</u> ptions	<u>H</u> elp	
2020-000000			2 5 5 6 6 5 5 6 5 5 6 6 6 6 6 6 6 6 6 6
SDSF SYSLOG 352 1	02 ULM4 ULM4 01/12/2	2009 OW 169 NO F	RESPONSE RECEIVED
	S CTZCIM		SCRULL ===> CSR
	012 15 02 28 27	00000000 TEO95	
N 0200000 WLM4 09	012 15:03:38.27 STC	0435 00000090 \$HASE	250 JESWIRDS PURGED
NR0000000 WLM4 09	012 15:12:23.17 INTE	RNAL 00000090 \$HASE	881 LOGON1 APPLI
NR0000000 WLM4 09	012 15:12:23.17 INTE	RNAL 00000090 SHASE	880 LINE80 UNIT
NR0000000 WLM4 09	012 15:12:23.18 INTE	RNAL 00000090 SHASP	679 \$SN, A=IPVATNJE
SR		RC=08	1.0
NR0000000 WLM4 09	012 15:12:23.18 INTE	ERNAL 00000090 \$HASP	670 \$SN,SOCKET=SAP9
SR		A VA	ALID IPADDR, RC=15
NC0000000 WLM4 09	012 15:12:23.18 INTE	RNAL 00000290 D ASM	1
MR0000000 WLM4 09	012 15:12:23.18 INTE	RNAL 00000090 IEE20	00I 15.12.23 DISPLAY
		998 00000090 TYPE	FULL STAT DEV
		998 00000090 PLPH	22% UK 5414
		998 00000000 LOCAL	0% OK 5414
ER		998 00000000 PAGE	TEL COMMAND IS NOT A
	012 15:19:16.41 INST	REAM 00000290 1000	
N 0200000 WIM4 09	012 15:19:37.17 TSU	0438 00000291 \$HASE	100 PRAK1 ON TSO
N 4000000 WLM4 09	012 15:19:37.21 TSUG	0438 00000090 SHASE	373 PRAK1 STARTE
N 0000000 WLM4 09	012 15:19:37.21 TSUG	0438 00000090 IEF12	5I PRAK1 - LOGGED O
N 0000000 WLM4 09	012 15:19:37.78	00000291 IEF19	6I IEF237I 5AA1 ALL
N 0000000 WLM4 09	012 15:19:37.78	00000291 IEF19	06I IEF285I WLM.BH
N 0000000 WLM4 09	012 15:19:37.78	00000291 IEF19	061 IEF2851 VOL SE
N 0000000 WLM4 09	012 15:19:41.44	00000291 IEF19	6I IGD103I SMS ALLO
N 0000000 WLM4 09	012 15:23:07.16 STC	00413 00000090 IEF40	4I BPXAS - ENDED -
N 0000000 WLM4 09	012 15:23:07.16 STC	00412 00000090 IEF40	4I BPXAS - ENDED -
N 0000000 WLM4 09	012 15:23:07.16 SIC	0410 00000090 IEF40	41 BPXAS - ENDED -
N 4000000 WLM4 09	012 15:23:07.16 5100		395 BPXHS ENDED
N 4000000 WLM4 09	012 15.23.07.16 8100		
	012 15.23.07.18	000000000 00000000000000000000000000000	AT SLID TRAD TD=X33
N 0000000 WIM4 09	012 15:23:07.18	00000290 TEA95	9T SLIP TRAP ID=X33
N 0000000 WLM4 09	012 15:23:07.18	00000290 IEA98	91 SLIP TRAP ID=X33
N 0200000 WLM4 09	012 15:23:07.19 STC	0413 00000090 \$HASE	250 BPXAS PURGED
N 0200000 WLM4 09	012 15:23:07.19 STCC	0410 00000090 SHASE	250 BPXAS PURGED
N 0200000 WLM4 09	012 15:23:07.19 STC	00412 00000090 \$HASF	250 BPXAS PURGED
N 0000000 WLM4 09	012 15:28:13.34 TSUG	00438 00000290 IEA63	10I OPERATOR PRAK1
F1=HELP F2=SPL	IT F3=END	F4=RETURN F5=IFIN	ID F6=BOOK
F7=UP F8=DOW	N F9=SWAP F	-10=LEFT F11=RIGH	II F12=RETRIEVE
M <u>A</u> a		Û	04/030
Connected to remote server/host tn3270.de.ib	m.com using lu/pool FU0U3104 and port 23		10

- 17. Wiederum kommt eine **No Response Received** Meldung und scheinbar geschieht nichts.
- Bevor man nun weitermacht, sollte vorher auf jeden Fall verifiziert werden ob die einzelnen Komponenten auch erfolgreich gestartet wurden. Dies wird mit dem Befehl '/d(isplay) a,l' erledigt.

Seite 16 von 50

과 Session A - [43 x 80]					_	
File Edit View Communication	Actions Window ZipPrin	nt Help				
		o o 📄 🛋 🔌				
<u>D</u> isplay <u>F</u> i	lter <u>V</u> iew	<u>P</u> rint <u>O</u> pti	ions <u>H</u> elp	>		
			40,000			
COMMOND INDUIT	352 142 1	WLM4 017	12/2009 0	W 165	NO RESPONSE RECEIVE	CD .
N 400000 WI MA		63.38 26	STC00435	000000000	SURVEL	IDED
N AAAAAAA WLM4	09012 1	5.03.38 27	01000400	000000000	TEA989T SLIP TRAP TO	1=X33
N 0200000 WLM4	09012 1	5.03.38 27	STC00435	00000250	SHASP250 JESWIRDS PL	IRGED
NRAAAAAAA MIM4	09012 1	5.12.23 17	INTERNAL	000000000	SHASP881 LOGON1 6	APPL T
NROOOOOO WLM4	09012 1	5.12.23 17	INTERNAL	000000000	SHASP880 LINERO	UNTT
NROOOOOO WLM4	09012 1	5.12.23 18	INTERNAL	000000000	SHASP679 SSN A=TPVAT	N.TE
SR	05011 1	0.12.120.10	2111 EIGHTE	00000000	RC=08	110 -
NR0000000 WLM4	09012 1	5:12:23.18	INTERNAL	00000090	\$HASP670 \$SN.SOCKET=	SAP9
SR					A VALID TRADDR. RC=	15
NC0000000 WLM4	09012 1	5:12:23.18	INTERNAL	00000290	DASM	
MR0000000 WLM4	09012 1	5:12:23.18	INTERNAL	00000090	IEE2001 15.12.23 DIS	PLAY
LR			998	00000090	TYPE FULL STAT	DEV
DR			998	00000090	PLPA 22% OK	5414
DR			998	00000090	COMMON 0% OK	5414
DR			998	00000090	LOCAL 0% OK	5514
ER			998	00000090	PAGEDEL COMMAND IS N	OT A
NC0000000 WLM4	09012 1	15:19:16.41	INSTREAM	00000290	LOGON	
N 0200000 WLM4	09012 1	15:19:37.17	TSU00438	00000291	\$HASP100 PRAK1 ON	I TSO
N 4000000 WLM4	09012 1	15:19:37.21	TSU00438	00000090	\$HASP373 PRAK1 ST	ARTE
N 0000000 WLM4	09012 1	15:19:37.21	TSU00438	00000090	IEF125I PRAK1 - LOGO	ED O
N 0000000 WLM4	09012 1	15:19:37.78		00000291	IEF196I IEF237I 5AA1	ALL
N 0000000 WLM4	09012 1	15:19:37.78		00000291	IEF196I IEF285I WL	M.BH
N 0000000 WLM4	09012 1	15:19:37.78		00000291	IEF196I IEF285I VC	L SE
N 0000000 WLM4	09012 1	15:19:41.44		00000291	IEF196I IGD103I SMS	ALLO
N 0000000 WLM4	09012 1	15:23:07.16	STC00413	00000090	IEF404I BPXAS - ENDE	D -
N 0000000 WLM4	09012 1	15:23:07.16	STC00412	00000090	IEF404I BPXAS - ENDE	D -
N 0000000 WLM4	09012 1	15:23:07.16	STC00410	00000090	IEF404I BPXAS - ENDE	D -
N 4000000 WLM4	09012 1	15:23:07.16	STC00413	00000090	\$HASP395 BPXAS EN	IDED
N 4000000 WLM4	09012 1	15:23:07.16	STC00412	00000090	SHASP395 BPXAS EN	IDED
N 4000000 WLM4	09012 1	15:23:07.16	81000410	00000090	SHASP395 BPXAS EN	IDED
N 000000 WLM4	09012 1	15:23:07.18		00000290	IEA9891 SLIP TRAP IL	=×33
N 000000 WLM4	09012 1	15:23:07.18		00000290	IEA9891 SLIP TRAP IL	= ×33
N 0000000 WLM4	09012 1		97000449	00000290	TEHSOSI SLIP IRAP IL	-X33
N 0200000 WLM4	09012 1	15.23.07.19	STC00413	00000090	CHASP250 BPXHS PURGE	D
N 0200000 WLM4	09012 1	15.23.07.19	STC00410	00000090	CHASP250 BPAHS PURGE	
N 0200000 WLM4	09012 1	15.28.12 24	TCU00412	000000000000000000000000000000000000000	TEACOOT ODEBATOD DE	
	E2=SDITT	E3=END	F4=D			
E7=LIP	F8=DOWN	F9=SHAD	F10=1	EFT F11	ERIGHT F12=RETRIE	VE
				11.		1007
			U		04	7027
Connected to remote server/h	ost tn3270.de.ibm.com using	g lu/pool FU0U3104 and por	t 23			11

19. Falls alles erfolgreich gestartet wurde, sollten Einträge wie **RMF**, **RMFGAT**, **GPMSERVE** und **CFZCIM** existieren.

Seite 17 von 50

TH Service A 142	. 901							
File Edit View Com	wunication Actions V	Vindow ZinPrint Heln						
<u>D</u> isplay	<u>F</u> ilter	<u>v</u> iew <u>P</u> ri	nt <u>Upt</u>	ons <u>H</u> elp	, 			
SDSF SYSL	OG 352	.102 WLM4	WLM4 01/	12/2009 0)W 169	COMMAND	ISSUED	
COMMAND I	NPUT ===>					8	CROLL ==	=> CSR
IFF114I	16.14.24	2009.012 A	CTIVITY	472				
JOBS	M/S	IS USERS	SYSAS	INITS	ACTIVE/	MAX VTAM	OAS	
00002	00026	00001	00031	00045	00001/0	0125	00015	
CTTX	CTTX	CTTX	NSW S	ZTTX	ZTTX	ZTTX	NSW S	
TN3270	TN3270	TN3270	NSW SC	D OAM	OAM	IEFPROC	NSW S	
RAGE	RAGE	RAGE	NSW SU			LLA	NSW S	
ASCH	ASCH	ASCH		FEST	FEST	FEGT		
ENOMON	FNOMON	FNOMON	OWT S	JES2	JES2	TEEBBOC	NSW S	
RRS	RRS	RRS	NSW S	NET	NET	NET	NSW S	
TSO	TSO	STEP1	OWT S	TCPIP	TCPIP	TCPIP	NSW SO	<u>ĝ</u>
INETD1	STEP1	OMVSKERN	OWT AC) FTPS1	STEP1	STCUSER	OWT AO	6
D941MST	R D941MST	R IEFPROC	NSW S	D941IRLN	1 D941IRLM		NSW S	
D941DBM	1 D941DBM:	I IEFPROG	NSW S	D941DIS	D941DIST	IEFPROG	NSW SO	6
PURIMHP	PURIMHP	PMEGAT	NEL SC			STED1		
CEZCIM	CEZCIM	*OMVSEX		DEMOENT		SILFI	NOW 00	8
VM	VM	VM	NSW S	00024K	- 00088K			
PRAK1	IN							
N 0000000	WLM4	99012 15:1	9:37.78		00000291	IEF196I	IEF237I	5AA1 ALL
N 0000000	WLM4	09012 15:1	9:37.78		00000291	IEF1961	IEF2851	WLM.BH
N 0000000	WLM4 (99012 15:1	9:37.78		00000291	1EF1961	1EF2851	VOL SE
N 0000000		0012 15:1	9:41.44	91000419	00000291	IEF1961		SMS HLLU
N 00000000		99012 15.2	3.07 16	STC00413	0000000000	1EF4041	BPXAS -	ENDED -
N 0000000	WLM4 (9012 15:2	3:07.16	STC00410	00000090	IEF4041	BPXAS -	ENDED -
N 4000000	WLM4 (99012 15:2	3:07.16	STC00413	00000090	\$HASP395	BPXAS	ENDED
N 4000000	WLM4 (99012 15:2	3:07.16	STC00412	00000090	\$HASP395	BPXAS	ENDED
N 4000000	WLM4 (09012 15:2	3:07.16	STC00410	00000090	\$HASP395	BPXAS	ENDED
N 0000000	WLM4 (09012 15:2	3:07.18		00000290	IEA989I	SLIP TRA	P ID=X33
N 0000000	WLM4 U	99012 15:2	3:07.18		00000290	IEA9891	SLIP IRA	P ID=X33
N 0200000		99012 15:2	3 07 19	STCO0413	00000250	\$HASP250	BDXAS D	URGED
N 0200000		99012 15.2	3:07.19	STC00410	00000090	SHASP250	BPXAS P	URGED
N 0200000	WLM4 (9012 15:2	3:07.19	STC00412	00000090	\$HASP250	BPXAS P	URGED
N 0000000	WLM4	99012 15:2	8:13.34	TSU00438	00000290	IEA630I	OPERATO	R PRAK1
F1=HELP	F2=SI	PLIT F	3=END	F4=RE	ETURN F	5=IFIND	F6=B0	OK
F7=UP	F8=D0	DWN F	9=SWAP	F10=LE	EFT F1:	I = RIGHT	F12=RE	TRIEVE
M <u>A</u> a				Û				04/021
Connected to remot	te server/host tn3270.d	e.ibm.com using lu/pool	FU0U3104 and por	t 23				1.

Wenn die einzelnen Komponenten zum ersten Mal gestartet werden sollen, dann könnte es vorkommen, dass die UserID keine ausreichenden Rechte besitzt. Dies kann an den entsprechenden Fehlermeldungen im LOG erkannt werden und muss von einem Admin im RACF gelöst werden.

1.3 Starten der DB2-Datenbank

Als nächstes sollte man verifizieren, dass die Datenbank läuft. Das **WLP-Framework** benötigt wegen den neu eingeführten **PureXML**-Technologie IBMs **DB2 for z/OS** in der Version 9.1 oder neuer.

In der Regel wird ein Datenbank-Administrator sich um das Aufsetzen der Datenbank kümmern. Falls er nach Details für die Installation fragen sollte: Die Verbindung zur Datenbank wird über JDBC hergestellt, eine Standard-Installation sollte vollkommen ausreichen (Die benötigten XML-Features werden per Default installiert). Da Tabellen angelegt werden müssen, benötigt man auch Admin-Rechte und zusätzlich ist es äußerst praktisch, wenn die **DB2 automatisch beim IPL gestartet** wird. All dies kann ein Datenbank-Administrator recht schnell einrichten.

20. Mit dem bekannten Befehl **'/d a,I'** kann im Log überprüft werden ob DB2 läuft. Es sollten Einträge mit dem Namen der DB-Instanz vorhanden sein. Diesen erhaltet ihr vom Datenbank-Administrator, ihr werdet ihn auch später noch für die Verbindung zur Datenbank benötigen.

Seite 18 von 50

Bession A - [43	x 80]							
File Edit View Com	munication Actions Wi	indow ZipPrint Help						
<u>D</u> isplay	<u>F</u> ilter	<u>V</u> iew <u>P</u> rir	nt <u>O</u> pti	ons <u>H</u> elp	`			
SDSE SYSI	06 352	102 WIM4 4	JI M4 017	12/2009 6)ผ 169	COMMAND	ISSUED	
COMMAND I	NPUT ===>			1212005 0		S	CROLL ==	=> CSR
RESPONSE=	WLM4							492 - 200 AF 13 - 2 - 2 - 3
IEE114I	17.25.37 2	009.012 AC	TIVITY	823				
JOBS	M/S T	S USERS	SYSAS	INITS	ACTIVE/N	1AX VTAM	OAS	2010-
CTTV	00027	CTTY	00032	7777	7777	7777	NGU 9	
TN3270	TN9270	TN3270	NSW SO			TEEBBOC	NSW S	
RACE	RACE	RACE	NSW SO	LLA	LLA	LLA	NSW S	
VLF	VLF	VLF	NSW S	APPC	APPC	APPC	NSW S	
ASCH	ASCH	ASCH	NSW S	FFST	FFST	FFST	NSW S	
ENQMON	ENQMON	ENQMON	OWT S	JES2	JES2	IEFPROC	NSW S	
RRS	RRS	RRS	NSW S	NET	NET	NET	NSW S	8
	180	STEP1	OWI S	TCPIP	ICPIP	TCPIP	NSW SO	
D941MST		TEEPROC	NGLI S	D941TRLM				
D941DBM	1 D941DBM1	IFFPROC	NSW S	D941DIST	D941DIST	IFFPROC	NSW SO	
PURIMAP	PURIMAP	PMAP	UWI SU	RMF	RMF	LEFPRUG	NSW S	-
RMFGAT	RMFGAT	RMFGAT	NSW SO	GPMSERVE	GPMSERVE	STEP1	NSW SO	5
CFZCIM	CFZCIM	*OMVSEX	IN SO	CAZO	CAZO	CAZO	NSW S	
VM	VM	VM	NSW S	00024K	- 00088K			
PRAK1		9619 15.10			00000001	TEELOCT	TEEDODT	
N 0000000		9012 15.15	3 37 78		00000291	1EF1961	1EF2851	LI M RH
N 0000000	WLM4 0	9012 15:19	37.78		00000291	IEF1961	IEF2851	VOL SE
N 0000000	WLM4 0	9012 15:19	9:41.44		00000291	IEF196I	IGD103I	SMS ALLO
N 0000000	WLM4 0	9012 15:23	3:07.16	STC00413	00000090	IEF4041	BPXAS -	ENDED -
N 0000000	WLM4 0	9012 15:23	3:07.16	STC00412	00000090	IEF404I	BPXAS -	ENDED -
N 0000000	WLM4 0	9012 15:28	3:07.16	STC00410	00000090	1EF4041	BEXAS -	ENDED -
N 4000000		9012 15:22	3:07.16	STC00413	00000090	\$HASP395	BPXAS	ENDED
N 4000000		9012 15.23	3.07 16	STC00412	000000090	\$HASP395	BEXAS	ENDED
N 0000000	WLM4 0	9012 15:23	3:07.18	01000410	00000290	IEA9891	SLIP TRA	P ID=X33
N 0000000	WLM4 0	9012 15:23	3:07.18		00000290	IEA989I	SLIP TRA	P ID=X33
N 0000000	WLM4 0	9012 15:23	3:07.18		00000290	IEA989I	SLIP TRA	P ID=X33
N 0200000	WLM4 0	9012 15:23	3:07.19	STC00413	000000000	\$HASP250	BPXAS P	URGED
N 0200000	WLM4 0	9012 15:23	3:07.19	STC00410	00000090	\$HASP250	BPXAS P	URGED
N 0200000		9012 15:22	3:07.19	51000412	00000090	5HASP250	BPXAS P	
F1=HFLP	F2=SP	ITT ES	3=FND	F4=RF		SETETND	E6=B0	OK
F7=UP	F8=D0	WN F9	9=SWAP	F10=LE	FT F11	I=RIGHT	F12=RE	TRIEVE
MALa				Ŷ				04/021
Connected to remov	te server/host tn3270 de	ibm.com.using.lu/pool.E	U0U3104 and port	23				
Connected to remo	12 22. Yel Jilose a 1527010C	and a string to poor t	eres to rand pore					10

Im vorliegenden Fall existieren Einträge für die DB-Instanz 'D941DBM1'.

Damit der **RMFRecorder** und die **WLPrediction** problemlos funktionieren, müssen die in der Datei **WLPDefaults.properties** (s. Kapitel 1.4) angegebenen Tabellen vor dem Ausführen der WLP-Anwendungen schon vorhanden sein.

Auf die Datenbank kann je nach Präferenz über ISPF und **SPUFI** oder mit Tools wie dem **IBM Data Studio Developer** und **SQuirreL** zugegriffen werden. Details zu den letzten beiden Tools folgen in Kapitel 2. Sobald eine Verbindung mit einem der Tools hergestellt wurde, sollten als erstes sämtliche benötigten Tabellen angelegt werden.

1.4 WLPDefault.properties

Die Konfiguration des **WLP-Frameworks** wird in erster Linie über die Datei **WLPDefaults.properties** im Root-Verzeichnis des WLPrediction-Projekts vorgenommen. Bevor also die Anwendungen gestartet werden können, sollten zuerst die benötigten Angaben in der Konfigurations-Datei gemacht werden.

Diese werden nachfolgen möglichst allgemein angegeben. Die in der Diplomarbeit verwendeten Daten werden als Beispiel in Klammern angegeben.

- Database configuration:
 - dbTrainTestURL = jdbc:db2://Host:Port/Db2-Instanz (jdbc:db2://BOEWLM4:5941/WLMD941)

Seite 19 von 50

- o dbTrainTestUser = Host-Login (prak1)
- o dbTrainTestPasswd = Host-Passwort (weihn8en)
- o dbOnlineURL = jdbc:db2://Host:Port/Db2-Instanz (jdbc:db2://BOEWLM4:5941/WLMD941)
- o dbOnlineUser = Host-Login (prak1)
- o dbOnlinePasswd = Host-Passwort (weihn8en)
- Which reader type to use:
 - ReaderType = **CIMReader**
- CIMReader settings:
 - cimomHost = host.boeblingen.de.ibm.com
 (boewlm4.boeblingen.de.ibm.com)
 - o cimomPort = Port (5988)
 - o cimomUser = Host-Login (prak1)
 - o cimomPass = Host-Passwort (weihn8en)
 - o cimMetricsFile = cimmetrics.xml
- Set query interval length:
 - o queryIntervalLength = 100000
- Specify if RmfInterval Length is used:
 - o useRmfIntervalLength = true
- Set query lag time:
 - o queryLagTime = 10000
- Downsampling factor:
 - downSamplingFactor = 5
- WLP Main Settings:
 - o wlpAgentRepositoryPath = AgentPool
 - asynchronProcessing = true
 - o srcTable = metrics (t060619a_ds5_xml)
 - o dstTable = prediction
- Web interface:
 - o wlpServerPort = 8081

Zu manchen Parametern finden sich in der Datei genauere Informationen.

1.5 BSO Authentication

Es muss beachtet werden, dass für den Zugriff innerhalb des IBM-Netzwerkes eine **BSO HTTPS Authentication** mit der **IntranetID** durchgeführt werden muss, da ansonsten kein Zugriff auf die Datenbank und den CIM-Server möglich ist. Es erscheint dann eine

Seite 20 von 50

entsprechende Fehlermeldung. Dies muss in einem bestimmten Zeitintervall (mehrere Stunden) wiederholt werden!

Seite 21 von 50

2 Herstellen einer Datenbankverbindung

Vor der Ausführung der WLP-Anwendungen müssen zunächst die benötigten Tabellen in der Datenbank erstellt werden. Das kann wahlweise über ISPF und SPUFI erfolgen oder bequem über einen graphischen SQL-Client. Im Anschluss werden zwei mögliche Werzeuge vorgestellt und alle zur Verbindungseinrichtung nötigen Schritte erläutert.

Ein Problem stellt der JDBC-Treiber für DB2 for z/OS dar. Es ist nicht möglich ihn isoliert zu beziehen. Er wird beispielsweise mit der Installation des im nächsten Kapitel vorgestellten Data Studio Developer mitinstalliert, welches als kostenlose Trial-Version öffentlich zugänglich ist.

2.1 IBM Data Studio Developer

IBMs Data Studio Developer (im Folgenden als **Data Studio** bezeichnet) ist eine IDE für Datenbankentwickler, die auf Eclipse basiert. Sie ist für Entwickler und Administratoren gleichermaßen konzipiert und bietet für beide Gruppen entsprechende Funktionalität.



Seite 22 von 50

2.1.1 Einrichten der Verbindung

Als erstes sollte mit dem Einrichten der Verbindung begonnen werden.

21. Im geöffneten Data Studio rechtsklick auf **Verbindungen** in der **Datenbank-Explorer** View und **Neue Verbindung...** auswählen



22. Es öffnet sich ein neues Fenster mit dem Titel Neue Verbindung. Hier unter Verbindungskennung den Haken bei Standardnamenskonvention verwenden belassen oder den Namen der Datenbank-Instanz als Verbindungsnamen eingeben. Als nächstes unter Datenbankmanager auswählen den Eintrag DB2 für z/OS aufklappen und Alle Versionen auswählen. Unter JDBC-Treiber den per Default ausgewählten IBM Data Server Driver for JDBC and SQLJ nicht verändern und als nächstes die URL-Verbindungsdetails spezifizieren: Als Speicherposition Namen der Datenbank-Instanz festlegen, den Hast und die Detnummer apgrifizieren und als latetee unter Klappennegitier

den **Host** und die **Portnummer** spezifizieren und als letztes unter **Klassenposition** den Pfad zum Jar-File des Treibers inklusive der Lizenz-Datei. In der Regel sollte das der Pfad **C:\Program**

Files\IBM\DS12Shared\plugins\com.ibm.datatools.db2_1.0.201.v200807111732\dr iver\ sein und die benötigten Jars sind db2jcc.jar und db2jcc_license_cisuz.jar. Zur Kontrolle muss im Feld JDBC-Treiberklasse der Eintrag

Seite 23 von 50

com.ibm.db2.jcc.DB2Driver stehen.

Abschließend noch die benötigten Benutzerinformationen **Benutzer-ID** und **Kennwort** eingeben und mit Klick auf den Button **Verbindung testen** die eingetragenen Angaben auf Korrektheit verifizieren.

Neue Verbindung				
Verbindungsparameter Wählen Sie den Datenbankmanager, de	n JDBC-Treiber und	d die erfi	orderlichen Verbindungsparameter aus.	_
Verbindungskennung Standardnamenskonvention verweiter verbindungsname: WLMD941	enden			
Datenbankmanager aus <u>w</u> ählen:	JDBC-Treiber:	IBM Data	a Server Driver for JDBC and SQLJ	
DB2 für Linux, UNIX und Windows	URL-Verbindur	ngsdetai	s	
Alle Versionen	Treiberoption	ien Tra	ceoptionen	
☐ DB2 für z/OS	Speicherposit	tion:	WLMD941	
Alle Versionen	Host:		BOEWLM4	
⊕ Informix	Portnummer:		5941	
	JDBC-Treiber	klasse:	com.ibm.db2.jcc.DB2Driver	
	Klassenpositi	on:	'111732\driver\db2jcc_license_cisuz.jar Durchs	uchen
	Nur Objek	te abru	fen, die von diesem Benutzer erstellt wurden	
	Verbindungs-	URL:	jdbc:db2://BOEWLM4:5941/WLMD941:retrieveMes sFromServerOnGetMessage=true;emulateParame taDataForZCalls=1;	ssage 🔨 terMe
Benutzerinformationen				
Kennwort: ******				
verbindung <u>t</u> esten	L			
(?)			< Zurück Weiter > Fertig stellen	Abbrechen

23. Falls mit einem Popup eine erfolgreiche Verbindung bestätigt wurde, auf **Fertigstellen** klicken um das Erstellen zu beenden.



2.1.2 Herstellen der Verbindung

Die neu eingerichtete Verbindung ist nun in der **Datenbank-Explorer** View verfügbar und kann eingesetzt werden.

24. Um eine Verbindung mit der Datenbank herzustellen muss man die Verbindung nur doppelklicken.

Über eine aufklappende Baum-Struktur sind dann alle vorhanden Daten wie Benutzer, Schemata, etc.

Unter Schemata befindet sich auch der jeweilige Benutzer und die von ihm erstellten Tabellen.

Seite 25 von 50



- 25. Nachdem die Verbindung hergestellt wurde, können sämtliche von Datenbanken bekannte Funktionen durchgeführt werden.
- 26. Entwickler können dann auch wie bei Eclipse üblich in der **Datenprojektexplorer** View verschiedene Projekte, so genannte **Datenentwicklungsprojekte** anlegen und komfortabel alles dafür benötigte (SQL-Skripte,...) anlegen

2.2 SQuirreL SQL-Client

SQuirreL ist ein universeller graphischer SQL-Client, der frei verfügbar ist. Mit SQuirreL ist

Seite 26 von 50

es möglich auf jede relationale Datenbank zuzugreifen, für die ein JDBC-Treiber verfügbar ist.

SQuirreL SQL Client Version 2.6.6		_ 🗆 🔀
Datei Ireiber Aliase Plugins Session Eenster Hilfe		
Verbinden mit: Wlp 🔻 🚓 📰 📰 🗊 🗗 🖛 Aktive Session:	- 👌 🖻 🛛	C 1
Treiber 🌵 🥢 🗈 🤎 😫 🦂 🚍 Aliase 📣 🌵 🖉 🗈 🕄 🛱 🖓		
X Apache Derby Client		
A Axion		
✓ DB2 for z/OS		
😵 Firebird JayBird		
Ø FrontBase		
H2 mbaddad		
Logs: Keine Fehler, Keine Warnung, 25 Nachrichten 📃 🖳 🛛 7 von 8 MB	23:0	0:24 CET

2.2.1 Hinzufügen und Laden des JDBC-Treibers für DB2 for z/OS

Für die erstmalige Benutzung muss der benötigte JDBC-Treiber **einmalig** hinzugefügt und geladen werden.

27. Im geöffneten SQuirreL-Fenster gibt es auf der linken Seite ein kleineres Fenster mit dem Titel **Treiber**. Um einen neuen Treiber hinzuzufügen, auf das +-Icon in der Titelleiste klicken.



28. Es öffnet sich ein neues Fenster mit dem Titel **Treiber Hinzufügen**. Hier als Name **DB2 for z/OS** und als Beispiel URL

Seite 27 von 50

jdbc:db2://<Server>:<Port>/<DB> spezifizieren.

Danach auf den Reiter Alternativer Class Path wechseln und auf den Button Hinzufügen klicken.

reiber			
Name:	DB2 for z/OS		
Beispiel URL:	jdbc:db2:// <server>:<port>/<db></db></port></server>		
Vebseite URL:			
lava Class Pati	Alternativer Class Path		
		A	Zeige Treiber an
			Rauf
			Runter
		-	Hinzufügen
L I		•	Löschen
assen Name:			
issen Name:			

29. Es öffnet sich ein weiteres Fenster in dem man den Pfad zum JDBC-Treiber angeben muss. Der mitgelieferte treiber befindet sich im Verzeichnis **DB2z-Treiber**. Dort mit Strg den Treiber **db2jcc.jar** und die Lizenz-Datei **db2jcc_license_cisuz.jar** auswählen und auf **Öffnen** klicken.

Seite 28 von 50

🍠 Öffnen		
Suchen <u>i</u> n:	DB2z-Treiber	- 🛍 🏠 🖽 🖿
 db2jcc.j db2jcc l 	ar icense cisuz.iar	
Datei <u>n</u> ame:	"db2jcc.jar" "db2jcc_license_cisuz.jar"	
Da <u>t</u> eityp:	JAR Dateien (*.jar, *.zip)	-
		Öffnen Abbrechen

Note: Falls das **Data Studio Developer** installiert wurde, dann wurden die benötigten DB2-Treiber mitinstalliert. Sie sie sollten sich im Verzeichnis **c:\Program Files\IBM\DS12Shared\plugins\com.ibm.datatools.db2_1.0.201.v200807111732\driver** befinden

TODO: Schauen ob bei DB2 Connect die Treiber auch installiert werden

 Zurück im Treiber Hinzufügen Fenster zuerst auf den Button Zeige Treiber an klicken, damit im Feld Klassen Name der Name com.ibm.db2.jcc.DB2Driver des JDBC-Treibers angezeigt wird. Zum Bestätigen auf den Button OK klicken.

Seite 29 von 50

Name:	DB2 for z/OS			
Beispiel URL:	jdbc:db2:// <server>:<port>/</port></server>	<db></db>		
/ebseite URL:				
character at	Albernahiuan Class Bath			
ava cidss Fau				
:\Documen <u>ts a</u>	nd Settings) Administrator (My D	A REAL PROPERTY AND A REAL		sine Trainer an
15	nd Secongs (Adminiscrator (My D	ocuments (Diplomarbeit (Di		eige freiber an
:\Documents a	nd Settings\Administrator\My D	ocuments (Diplomarbeit (Di ocuments (Diplomarbeit (Di	B2z-Tre	eige freiber an
:\Documents a	nd Settings\Administrator\My D	ocuments (Diplomarbeit (Di ocuments \Diplomarbeit \Di	32z-Tre	Rauf
:\Documents a	nd Settings\Administrator\My D	ocuments\Diplomarbeit\Df	322-Tre	Rauf Runter
:\Documents a	nd Settings\Administrator\My D	ocuments\Diplomarbeit\D		Rauf Runter Hinzufügen
:\Documents a	nd Settings\Administrator\My D	ocuments\Diplomarbelt\Df		Rauf Runter Hinzufügen Löschen
:\Documents a	nd Settings\Administrator\My D	ocuments (Diplomarbeit (Di		Rauf Runter Hinzufügen Löschen

31. Der benötigte JDBC-Treiber ist hiermit hinzugefügt und geladen. Wenn das Hinzufügen erfolgreich war, dann ist vor dem namen des hinzugefügten Treibers ein blauer Haken zu sehen.

	×
Treiber 🖶 🥖 🗈 🎱 🗱 🦂 🚔	
🔇 Apache Derby Client	٠
🔇 Apache Derby Embedded	
🔇 Axion	
✓ DB2 for z/O5	
🔇 Firebird JayBird	
8 FrontBase	
🔇 H2	
🔇 H2 Embedded	
🔇 H2 In-Memory	-

2.2.2 Einrichten der Verbindung

Nachdem der Treiber hinzugefügt und geladen ist, kann im nächsten Schritt die Verbindung zur Datenbank eingerichtet und im Anschluss hergestellt werden.

Seite 30 von 50



- 32. Im geöffneten SquirreL-Fenster gibt es auf der rechten Seite ein weiteres kleines Fenster mit dem Titel **Aliase**. Um einen neue Verbindung hinzuzufügen, auf das +-Icon in der Titelleiste klicken.
- 33. Es öffnet sich ein neues Fenster mit dem Titel Alias Hinzufügen. Hier als Name z.B. WLPDB1 spezifizieren. Im Dropdown-Menü Treiber den hinzugefügten JDBC-Treiber DB2 for z/OS auswählen. Im Feld URL das vorgegebene Template mit den entsprechenden Daten anpassen, im vorliegenden Fall wäre das jdbc:db2://BOEWLM4:5941/WLMD941. Optional können auch noch weitere Angaben gemacht werden, z.B. zum Benutzernamen und Passwort.

Bevor die Verbindung gespeichert wird, sollte sie getestet werden, dazu muss auf den Button **Test** geklickt werden

Name:	WLPDB1
Treiber:	✓ DB2 for z/OS ✓
URL:	jdbc:db2://BOEWLM4:5941/WLMD941
Benutzername:	prak1
Passwort:	
Auto Logon	Verbinden beim Starten Eigenschaften
Achtung! Passw	orte werden im Klartext gespeichert.

34. Es öffnet sich ein weiteres Fenster mit dem Titel **Verbinden mit: WLPDB1**. Nachdem Benutzernamen und Passwort eingegeben wurden auf den Button **Verbinden** klicken.

Seite 31 von 50

Bei erfolgreicher Herstellung der Verbindung wird dies mit einer entsprechenden Meldung angezeigt.

/erbinder	n mit: WLPDB1
Alias:	WLPDB1
Treiber:	DB2 for z/OS
URL:	jdbc:db2://BOEWLM4:5941/WLMD941
enutzer:	prak1
asswort:	
ala bu un a	Bas Ladulana bai dar Sisasha yan Dassyuttan d
	Lads Lock kann dei der Eindade von Passworten si
chicung - (· · · · · · · · · · · · · · · · · · ·
cncung - (Verbinden Abbrechen
rbir	Verbinden Abbrechen
erbir	Verbinden Abbrechen
entung - (Verbinden Abbrechen

(**Zu beachten**: Wenn die Herstellung der Verbindung nicht gelingt, dann kann das erneut an den Voraussetzungen aus Kapitel 1 liegen, vor allem 1.1, 1.3 und 1.5.)

35. Nach erfolgreichem Test das Status-Fenster mit **OK** schließen und die Verbindung mit Klick auf den **OK** Button speichern.

Name:	WLPDB1
Treiber:	✓ DB2 for z/OS ✓
URL:	jdbc:db2://BOEWLM4:5941/WLMD941
Benutzername:	prak1
Passwort:	
Auto Logon	🗌 Verbinden beim Starten
	📸 Eigenschaften
Achtung! Passw	orte werden im Klartext gespeichert.

36. Die Verbindung ist nun eingerichtet und steht im Fenster Aliase zur Verfügung.



2.2.3 Herstellen der Verbindung und kurze Einführung in die Benutzung von SQuirreL

Die neu eingerichtete Verbindung ist nun im Fenster **Aliase** verfügbar und kann eingesetzt werden.

37. Im Fenster Aliase auf die Verbindung WLPDB1 doppelklicken. Im sich öffnenden Fenster Verbinden mit: WLPDB1 die nötigen Angaben für Benutzer und Passwort eingeben und auf den Button Verbinden klicken um die Verbindung herzustellen.

Seite 33 von 50

Verbinder	1 mit: WLPDB1
Alias:	WLPDB1
Treiber:	DB2 for z/OS
URL:	jdbc:db2://BOEWLM4:5941/WLMD941
Benutzer:	prak1
Passwort:	•••••
	🖶 Eigenschaften
Achtung - (Caps Lock kann bei der Eingabe von Passworten störer
	Verbinden Schließen

38. Es öffnet sich ein neues Fenster mit der Datenbankverbindung für den gewählten Benutzer.

🧻 1 - WLPDB1 als prak1					
a 🕼 🖈 🤣	› h 🛛 🖳 🛆 🗶 f () i i 🔍 el 🧔	(1) (2)			
Objekte \ SQL \ Hibernate \					
		(Alterna) Call Scool and a)			
🗄 🗀 ADB	String-runktionen (System-runktionen (Zeit/Datums-runktionen (Schusseiworte (Mats Datum) (Datum) (Schusseiten Schusseiten (Schusseiten (Schusseiten (Schusseiten (Schusseiten (Schusseiten (S				
DB2OSC	Meta-Daterr / Status / Supported Reractorings / Sche	emaca (Tabellencypen (Datencypen) Numerische Funktionen			
DB2OSCA	Eigenschaftsname	Wert			
DSNACC	JDBC-Treiber Klassenname	com.ibm.db2.jcc.DB2Driver			
DSNRGCOL	JDBC-Treiber-Klassenpfad	C:\Documents and Settings\Administrator\My Documents\D			
	getDriverName	IBM DB2 JDBC Universal Driver Architecture			
	getDatabaseProductName	DB2			
	supportsCatalogsInTableDefinitions	raise			
ET STOPUN	supportsSchemasInTableDefinitions	true			
	getCatalogSeparator	r Falsa			
storesMixedCaseIdentifiers		tase			
	supportsSchemasInDataManinulation	true			
	supportsCatalogsInDataManipulation	false			
	getIdentifierOugleString	n			
	getDatabaseProductVersion	DSN09015			
	getDefaultTransactionIsolation	TRANSACTION READ COMMITTED			
	getMaxConnections	0			
	getResultSetHoldability	1			
	allProceduresAreCallable	false			
	allTablesAreSelectable	false			
	nullsAreSortedHigh	true			
	nullsAreSortedLow	false			
	nullsAreSortedAtStart	false			
	nullsAreSortedAtEnd	raise			
1	getDriverVersion	3.50.152			
/WLPDB1		1,1			

39. Mit Hilfe des Reiters **Objekte** in der linken Hälfte kann man durch die verschiedenen Datenbank-Objekte navigieren. Von Interesse ist aber nur das Verzeichnis mit dem Benutzer, im vorliegenden Fall **PRAK1**. Im Unterverzeichnis **Table** finden sich alle angelegten Tabellen.

Wenn eine Tabelle ausgewählt wurde, hat man auf der rechten Seite mehrere Reiter

Seite 34 von 50

mit verschiedenen Funktionen. Über den Reiter **Inhalt** kann man sich den Inhalt der Tabelle ansehen. Es finden sich auch allgemeine Informationen und Meta-Daten in den verschiedenen Reitern auf der rechten Seite.

🧻 1 - WLPDB1 als prak1		r ² ⊠
// * * * * * * * * * * * * * * * * * *	1. 🗁 🗶 🗘 🖓 🗧 🖓	옥 생 🔕 🐺 🕅 🖆
Objekte SQL \ Hibernate \		
E WLPDB1 ⊕ ⊡ ADB	Importierte Schlüssel \ Indiz Quellcode \ Info \ Inhal	es \Rechte \Spaltenrechte \Zeilen-IDs \Versionen \ Zeilenanzahl \ Spalten \ Primärschlüssel \ Exportierte Schlüssel \
E DB2OSC	TIMESTAMP	METRICSXML
DSNRGCOL DSNRGCOL DSNRGCOL DSNRGCOL DSNGCOL ALXIS GOBAL TEMPORARY TABLE GLOBAL TEMPORARY TABLE SYNONYM SYSTEM TABLE DSNGCOURE DSNGCOL DSNGCOL	1970-01-01 02:08:20.0 1970-01-01 02:16:40.0 1970-01-01 02:15:00.0 1970-01-01 02:25:00.0 1970-01-01 02:33:20.0 1970-01-01 02:58:20.0 1970-01-01 03:06:40.0 1970-01-01 03:15:00.0 1970-01-01 03:31:40.0 1970-01-01 03:40:00.0 1970-01-01 03:40:00.0 1970-01-01 03:56:40.0 1970-01-01 04:50:00.0 1970-01-01 04:21:40.0	<pre><monitoredmetrics><metric id="LPI"><instance id="LPI_BATCH3A.1 <MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3A.1 <MonitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3A.1 </monitoredMetrics><Metric ID=" lpi"=""><instance id="LPI_BATCH3A.1 </p></td></tr><tr><td>⊕ G UDF ⊕ SYSIBM ⊕ SYSIBM ⊕ SYSIBMTS ⊕ SYSFUN</td><td>1970-01-01 04:30:00.0
1970-01-01 04:38:20.0
1970-01-01 04:46:40.0
1970-01-01 04:55:00.0
1970-01-01 05:03:20.0</td><td><pre><monitoredmetrics><metric ID=" lp1"=""><instance <monitoredmetrics="" id="LP1_DATCH3A.1"><metric id="LP1"><instance id="LP1_BATCH3A.1 <MonitoredMetrics><Metric ID=" lp1"=""><instance id="LP1_BATCH3A.1 <MonitoredMetrics><Metric ID=" lp1"=""><Instance ID="LP1_BATCH3A.1 </pre></instance></instance></metric></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></instance></metric></monitoredmetrics></pre>
/WLPDB1/PRAK1/TABLE/T060619A_DS5_XML (8	35)	1,1

 Mit Hilfe des Reiter SQL kann man SQL-Abfragen auf die Datenbank ausführen. Dabei kann man mit Hilfe von verschiedenen Reitern in der unteren Hälfte auch die Ergebnisse mehrerer Abfragen parallel vergleichen.
 Für jede Abfrage stehen neben den Ergebnissen auch Meta-Daten und weitere Informationen zur Verfügung.

Seite 35 von 50

🧻 1 - WLPDB1 als prak1					් රේ 🗵
2 章 末 🕸 🕄	😕 🔁 🖪 🖳 🎂 🗶 🗘 🖟	j 🔍 🛃 🧔	🐺 🕅 🗗		
Objekte SQL Hibernate \					
Select * from prediction				🕶 🖡 🔲 Zeile	nlimit: 100
Select * from predicti	ion				57 E.
AT					
Select * from m \ Select * fro	mp				
Select * from prediction					র* দে" 🕅
Ergebnicce (Mathedates) Tel					
Ergebnisse (Metadaten) In	ro 1				
TIMESTAMP	TYPE		PI		
1970-01-01 02:00:00.0	LPI_BATCH2A.1		1,85		
1970-01-01 04:46:40.0	LPI_BATCH2A.1_prediction:WlpAgent		0,93865		333
1970-01-01 02:00:00.0	LPI_BATCH2A.1_target:WlpAgent		1,13853		
1970-01-01 02:08:20.0	LPI_BATCH2A.1		0,77		
1970-01-01 04:55:00.0	LPI_BATCH2A.1_prediction:WlpAgent		0,93865		
1970-01-01 02:08:20.0	LPI_BATCH2A.1_target:WlpAgent		1,13853		
1970-01-01 02:16:40.0	LPI_BATCH2A.1		0,81		
1970-01-01 05:03:20.0	LPI_BATCH2A.1_prediction:WlpAgent		0,93865		
1970-01-01 02:16:40.0	LPI_BATCH2A.1_target:WlpAgent		1,13853		
1970-01-01 02:25:00.0	LPI BATCH2A.1		1,11		
1970-01-01 05:11:40.0	LPI BATCH2A.1 prediction:WlpAgent		0.93865		
1970-01-01 02:25:00.0	LPI BATCH2A.1 target:WlpAgent		1.13853		
1970-01-01 02:33:20.0	LPI BATCH2A.1		1.64		
1970-01-01 05:20:00.0	LPI_BATCH2A.1_prediction:WlpAgent		0,93865		•
/ /WLPDB1/PRAK1/TABLE/T0606	19A_DS5_XML (835)				1,25

3 Benutzung des ABLE-Editors (Projekt Ablegui)

Der ABLE-Editor ist eine Anwendung mit graphischer Benutzeroberfläche, um Agenten für die Vorhersage zu erstellen, zu konfigurieren und zu Trainieren bevor man sie in der eigentlichen WLPrediction-Anwendung einsetzen kann.

3.1 WIpAgent erstellen

41. Der ABLE-Editor wird mit der Datei **startAbleGui.bat** im Root-Verzeichnis des WLPrediction-Projektes gestartet. Nach dem Start öffnet sich folgendes Fenster:



42. Für das WLP-Framework wurde ABLE um einige Beans erweitert. Aus diesen Beans wurde ein neuer Agent zusammengesetzt, der WIpAgent. Dieser Agent beinhaltet automatisch alle benötigten Beans. Man kann ihn durch einen Linksklick auf File → New agent → com.ibm.wIp.able.WIpAgent erstellen.

Seite 37 von 50

ile Edit View Tools V	Vindows Help				
New agent Open agent Save agent Save agent as	Default (com.ibm.able.AbleDefaultAgent) com.ibm.able.agents.AbleRuleAgent com.ibm.able.agents.AbleGeneticSearchAgent com.ibm.able.beans.petrinet.AblePetriNet	ne	Conversation	Platform	
Import bean Import remote agent Export bean	com.ibm.able.agents.AbleScriptAgent com.ibm.able.autotune.AutotuneAgent com.ibm.able.agents.AbleJavaScriptAgent com.ibm.able.conversation.AblePlatformConversationAgent				
Preferences	com.ibm.able.platform.AblePlatformDefaultAgent				
Printer setup	com.ibm.able.examples.rules.SANStorageManagementAgent com.ibm.able.examples.ableagent.SimpleAbleAgent				
Prinkso	com.ibm.wlp.able.WlpAgent				
Exit					

43. Im linken Bereich sollte nun ein **WIpAgent** vorhanden sein. Diesen muss man nun rechts-klicken und **Properties...** auswählen.

📓 Able Editor: untitled.ser
File Edit View Tools Windows Help
Data Learning Rules Agents Samples PetriNet Autotune Conversation Platform
WipAgent Selected Agent: WipAgent
WipAgent Inspect
Properties
Data flow
Timer events
Reset
Process
User functions

Seite 38 von 50

44. Es öffnet sich das Fenster **WIpAgent properties**, in welchem man den WIpAgent konfigurieren kann. Um das Erstellen des WIpAgents fertigzustellen müssen zuerst die restlichen benötigten Beans generiert werden.

Dazu wird auf den Reiter **Data Sources** gewechselt um dort sämtliche benötigten Daten zu spezifizieren.

🗟 WIpAgent properties 🛛 🔀							
General Data Source	Filter Network Co	onnections Functi	ons				
	Database URL:		Table:	User:	Passwd:		
Train/Test data:	jdbc:db2://BOEWLM4	:5941/WLMD941	'a_ds5_xml	prak1	veihn8en		
Online data:	jdbc:db2://BOEWLM4:5941/WLMD941 a_ds5_xml pr				veihn8en		
MetricIds:	LPI						
MetricInstIds:	LPI_BATCH2A.1						
MetricIds Output:	LPI_BATCH2A.1						
Datasources	Begin	End		#Records			
Train:)-01-01 02:00:00.0			668	From Test		
Test:)-01-05 21:50:	00.0	167			
Online:)-01-01 02:00:00.0)-01-05 21:50:	00.0				
			ОК Са	ncel	Help		

Für die verwendeten Referenzdaten von Clemens Gebhard wären das:

- Train/Test/Online Data:
 - Database URL: jdbc:db2://BOEWLM4:5941/WLMD941
 - Table: **t060619a_ds5_xml**
 - User: Host-Login (prak1)
 - Passwd: Host-Passwort (weihn8en)
- Metriclds: LPI
- MetricInstIds: LPI_BATCH2A.1
- MetricIds Output: LPI_BATCH2A.1
- Train
 - Begin: **1970-01-01 02:00:00.0**
 - End: LEER LASSEN!
 - o #Records: 668

Seite 39 von 50
- Test
 - Begin: LEER LASSEN!
 - End: **1970-01-05 21:50:00.0**
 - o #Records: 167
- Online
 - Begin: **1970-01-01 02:00:00.0**
 - End: 1970-01-05 21:50:00.0
 - #Records: LEER LASSEN!

(**Note:** Bei den Datasources **Train/Test/Online** kann man drei Variablen angeben, aber es reichen auch jeweils zwei Stück aus, denn der WlpAgent kann aus den zwei angegebenen Werten den dritten errechnen)

45. Wenn alle Daten erfolgreich angegeben wurden, muss als nächstes auf den Reiter **Network** gewechselt werden. Sämtliche benötigten Voreinstellungen werden dem WlpAgent automatisch vorgegeben. Jetzt muss nur noch auf den Button **Generate Beans** geklickt werden, um sämtliche Beans des WlpAgents automatisch zu generieren.

Agent Model	Turin	Train/I	ect Pation	
Agent Houe.	Irain			10
Inpu	uts Hidden1	Hidden2 Hidden	3 Outputs	Feedback
Architecture:	30	30 0	0	None
Activation functions:	TAN	✓ TAN ✓ TAN	. 🕶 SAT	~
		Generate Beans		
Window Size:	40	Horizon:	20	
Tolerance:	0.0	PF-Err factor:	10.0	
Maximum AveRMSError:	0.1	PF-Err fixed:	0.1	
Maximum Passes:	100	PF-Err pivot:	1.4	
		Msereg factor:	0.9	
	Start	F	Reset Beans	

(**Note:** Da beim Generieren auf die Datenbank und somit auf das Intranet zugegriffen wird, müssen die **Voraussetzungen** unter Kapitel 1 erfüllt sein!)

Seite 40 von 50

- 46. Mit den angegebenen Daten wurde nun ein vollständiger, aber noch **untrainierter WIpAgent** erstellt. Dieser sollte aus den folgenden Komponenten bestehen:
 - TrainingImport
 - TestImport
 - OnlineImport
 - Backpropagation
 - InFilter
 - OutFilter
 - MovingAverageFilter
 - TimeSeriesFilter

3.2 WlpAgent trainieren

Der erstellte Agent muss nun für die Generierung von Vorhersagen trainiert werden. Auch dies geschieht mit Hilfe des ABLE-Editors.

47. Dazu wird der erstellte WlpAgent rechts-geklickt und Properties... ausgewählt. Anschließend muss erneut in den Reiter Network gewechselt werden. Wenn ein WlpAgent neu erstellt wird, ist er zu Beginn automatisch im Train Mode. Um das Trainieren nun zu starten, muss man auf den Button Start links unten klicken.

Seite 41 von 50

General Dat	ta Sources	Filter	Network	Conn	ections	Fur	nctions				
Agent Mod	de:		Train		~		Train/Tes	t Ratio:	10		
		Inputs	Hidde	n1	Hidden2		Hidden3	Outputs	F	eedback	
Architectu	ire:	0	30		30		0	0		None	~
Activation	functions:		TAN.		TAN	~	TAN	✓ SAT	~		
					Generate	e Be	eans				
Window Si	ize:	2	10		Hoi	rizo	n:	20			
Tolerance	Tolerance: 0.0		0.0	PF-Err fac			tor: 10.0				
Maximum AveRMSError: 0,		0.1	PF-Err fix		fixe	ed: 0.1					
Maximum Passes:		1	100	PF-Err pi		pivo	ot:	1.4			
					Msereg	fa	ctor:	0.9			
			Start				Re	set Beans			

Der Agent läuft jetzt über den angegebenen **Trainingsdatensatz**. Das kann je nach Datensatz mehrere Minuten dauern. Mit dem Datensatz von Clemens Gebhard dauerte es knapp über 15 Minuten. Wenn der Agent brauchbare Ergebnisse liefern soll, dann sollte er einen **avgRMSError** von **0,21** oder weniger haben.

48. Der ABLE-Editor stellt für die Überwachung und Kontrolle des Trainings so genannte **Inspectors** zur Verfügung. Diese Inspektoren gibt es sowohl in Textform als auch in graphischer Form. Ein Beispiel für einen sinnvollen **Inspector** auf das **Backpropagation Bean** könnte so aussehen:

Bean	Edit	Data	View	Options	Help	
avgR	MSEr	ror:	0.21	1038617	0495	68728
last	RMSE	rror	: 0.0	948918	8236	534885
maxR	MSEr	ror:	0.80	0114914	0208	0468
mean	Squa	reEr	ror:	0.0678	7663	454678787
netE	poch	: 53				
netR	ecIn	x: 50	88			
netS	teps	PerE	poch:	605		
reco	rdIn	dex:	588			
stdD	evia	tion	: 0.1	1537965	8749	100872

Seite 42 von 50

Es wird u.a. der durchschnittliche Fehler, die Standardabweichung, der aktuell bearbeitete Datensatz und die aktuell bearbeitete Epoche angezeigt.

49. Wenn der WlpAgent seine voreingestellte Anzahl von Epochen (per Default 100) abgearbeitet hat, ist das Training beendet und der Agent wird in den **Run Mode** gesetzt.

Das noch offene Properties Fenster kann jetzt mit Klick auf **OK** geschlossen werden.

3.3 WlpAgent speichern

50. Als letztes muss der Agent noch zur weiteren Verwendung in der WLPrediction gespeichert werden. Dazu wählt man unter File → Save agent as... einen Namen und den Speicherplatz

für den serialisierten Agenten aus. Als Speicherplatz bietet sich der in der Konfigurations-Datei spezifizierte Pfad zum **AgentPool** an.



Damit der WlpAgent zur Vorhersage benutzt werden kann, muss er erstens im **Run Mode** sein und zweitens muss er im **AgentPool** liegen. Beide Voraussetzungen müssen erfüllt sein!

Seite 43 von 50

4 Starten des RMFRecorders

Zur Ausführung des RMFRecorders müssen sämtliche Voraussetzungen aus Kapitel 1 erfüllt sein. Zusätzlich müssen die zur Verwendung festgelegten **Tabellen angelegt** sein. Der RMFRecorder wird mit dem Shell-Skript **startRMFRecorder.bat** im Root-Verzeichnis des WLPrediction-Projektes gestartet (U.U. muss man im Batch-Skript den Pfad zum Workspace und zur java.exe anpassen).

Der Aufruf in der .bat-Datei muss mit den folgenden Jar-Dateien erfolgen:

java.exe -cp

".;bin;lib/sblim/sblimCIMClient.jar;lib/db2/db2jcc.jar;lib/db2/db2jcc_license_cisuz.jar ;lib/jdom/jdom.jar;lib/jdom/jaxen-core.jar;lib/jdom/jaxenjdom.jar;lib/jdom/saxpath.jar" com.ibm.wlp.rmfrecorder.RMFRecorder

Man sollte darauf achten, dass alle benötigten, oben angegebenen Jar-Archive beim Aufruf angegeben werden. Beim fehlen eines Jar-Archivs wird eine entsprechende Fehlermeldung angezeigt.

Der RMFRecorder ist für das Aufzeichnen der Metrik-Daten zuständig. Dazu holt sich die Anwendung die RMF-Metriken in einem festgelegten Zeitintervall mit Hilfe eines CIM-Clients vom CIM-Server ab und speichert sie in einem XML-Dokument pro Zeitstempel in der Tabelle **srcTable** (Referenzdatensätze: **metrics**).

Seite 44 von 50

5 Starten der WLPrediction

Zur Ausführung der WLPrediction müssen sämtliche Voraussetzungen aus Kapitel 1 erfüllt sein (wobei Kapitel 1.2 hier keine Rolle spielt). Zusätzlich müssen die zur Verwendung festgelegten **Tabellen angelegt** sein, das **srcTable** Daten enthalten, die verarbeitet werden können und für das berechnen von Voraussagen muss der benutzte WIpAgent auch im **Run Mode** arbeiten.

Die WLPrediction wird mit dem Shell-Skript **startWLP.bat** im Root-Verzeichnis des WLPrediction-Projektes gestartet (U.U. muss man im Batch-Skript den Pfad zum Workspace und zur java.exe anpassen).

Der Aufruf in der .bat-Datei muss mit den folgenden Jar-Dateien erfolgen:

java.exe -cp

".;bin;lib/able/able.jar;lib/able/ablebeans.jar;lib/able/JLog.jar;lib/jetty/jetty-6.1.4.jar;li b/jetty/jetty-util-6.1.4.jar;lib/jetty/servlet-

api-2.5-6.1.4.jar;lib/charting/jCharts-0.7.5.jar;lib/db2/db2jcc.jar;lib/db2/db2jcc_licens e_cisuz.jar;lib/jdom/jdom.jar;lib/jdom/jaxen-core.jar;lib/jdom/jaxen-jdom.jar;lib/jdom/ saxpath.jar" com.ibm.wlp.prediction.WLPrediction

Man sollte darauf achten, dass alle benötigten, oben angegebenen Jar-Archive beim Aufruf angegeben werden. Beim fehlen eines Jar-Archivs wird eine entsprechende Fehlermeldung angezeigt.

Die WLPrediction berechnet die eigentliche Voraussage. Dazu lädt die Anwendung zunächst alle Datensätze (Train, Test und Online) aus der Tabelle **srcTable** (Referenzdatensätze: **t060619a_ds5_xml**) und lädt den gespeicherten **trainierten WIpAgent** aus dem **AgentPool**. Danach arbeitet sie jeden Eintrag des angegebenen Zeitintervalls einzeln ab und schreibt die Vorhersagewerte in die festgelegte Tabelle **destTable** (Referenzdatensätze: **prediction**) der Datenbank zurück.

Seite 45 von 50

6 Migration auf den Großrechner

Das WLP-Framework sollte mit **Eclipse/Rational** bearbeitet werden, da die folgenden Schritte auch darauf ausgelegt sind. Selbstverständlich kann die Entwicklung mit jedem beliebigen Werkzeug angegangen werden.

6.1 Vorgehensweise

In diesem kapitel sollen die nötigen Schritte zur Migration auf den Großrechner erklärt werden.

- 51. Als erstes muss das Projekt WLPrediction im Eclipse-Workspace neue Erstellt werden, damit alle Änderungen im Quellcode übernommen werden. Dazu das im Projekt-Root vorhandene Ant-Skript build.xml ausführen WICHTIG: Es sollte darauf geachtet werden, welche JRE auf dem verwendeten Großrechner installiert ist! Die installierte JRE sollte mindestens die selbe Version besitzen wie der benutzte Java-Compiler, d.h. bei einer installierten JRE 1.5 darf der Java-Compiler maximal in der Version 1.5 vorliegen, 1.6 würde nicht laufen.
- 52. Wenn alle Änderungen kompiliert wurden, dann kann das Verzeichnis dist, ein Unterverzeichnis im WLPrediction-Projekt, mit einem FTP-Client auf die USS des Großrechners übertragen werden. Dazu müssen zwei Punkte beachtet werden:
 1. Die Adresse für den Host muss bekannt sein (im vorliegenden Fall boewlm4) und man muss seinem FTP-Client mitteilen, dass das Startverzeichnis auf dem Host das eigene Benutzer-Verzeichnis ist (im vorliegenden Fall /u/prak1), denn nur dafür werden ausreichend Rechte besessen. Dieses Verzeichnis muss selbstverständlich auch existieren.

2. Beim Übertragen der Dateien muss man penibel darauf achten welche Datei in welchem **Übertragungsmodus** hochgeladen wird.

Gewöhnliche Textdateien wie HTML- und Skript-Dateien **müssen** im **ASCII-Modus** übertragen werden. Dabei wird dann automatisch eine echte Konvertierung in die von Großrechnern verwendete **EBCDIC-Zeichenkodierung** vorgenommen, und nicht nur eine Konvertierung der Zeilenumbruchsvarianten der verschiedenen Betriebssysteme.

Im Klartext heisst das für den vorliegenden Fall:

Alle Dateien (größtenteils sind das binäre Jar-Dateien) müssen binär übertragen werden ausser die Shell-Skripte startRMFRecorder und startWLP, sowie die Konfigurationsdatei WLPDefaults.properties.

Die XML-Dateien **cimmetrics.xml** und **cimmetrics.xsd** müssen **explizit** (obwohl sie Textdateien sind!) im **binären Modus übertragen** werden, da sonst der RMFRecorder nicht funktioniert. Diese tatsache mag verwundern, aber das liegt daran, dass das komplette **Projekt auf ASCII kompiliert** wurde und der **RMFRecorder** beim Einlesen der **cimmetrics.xml** deswegen eine **ASCII-Datei**

erwartet.

Diese These wurde aber noch nicht durch entsprechende Tests bestätigt! Das sollte nachgeholt werden.

Seite 46 von 50

- 53. Wenn dann alle benötigten Dateien korrekt übertragen wurden müssen nur noch die entsprechenden Rechte zum Ausführen der Start-Skripte **startRMFRecorder** und **startWLP** vergeben werden. Das erledigt man mit dem Befehl **chmod 755 startRMFRecorder** und **chmod 755 startWLP** respektive.
- 54. Schließlich sind der RMFRecorder sowie die WLPrediction auf ihre Ausführung vorbereitet und können mit den entsprechenden Start-Skripten gestartet werden. Genau wie bei der Ausführung auf dem Client, muss auch hier wieder auf die Voraussetzungen aus Kapitel 1 verwiesen werden.
 Wichtig: Durch ihre Architektur, laufen die Anwendungen in einer Endlosschleife, um sie zu beenden muss man das SIGINT-Signal abschicken, wie das auf dem Großrechner bewerkstelligt wird, wird im folgenden kapitel kurz beschrieben.

6.2 Beenden der WLP-Anwendungen auf dem Großrechner (SIGINT)

Auf dem Windows-Client wird das **SIGINT-Signal** mit der **Tastenkombination Strg + C** abgeschickt. Das wird auf dem Großrechner anderst erledigt. Eine mögliche Lösung wird im folgenen erläutert.

55. Wenn man sich in der **OpenMVS Shell** befindet, dann befindet sich links unten im Eck die Information **ESC=Ý**.



Seite 47 von 50

56. Da man dieses kryptische Zeichen auf der tastatur nur schwer findet, wird es mit Hilfe des Terminal Emulator einfach mit der Maus markiert und per **copy&paste** auf die Kommandozeile kopiert. Das geht wahlweise über das Menü oder die **zwei Buttons** links oben. Der linke kopiert und der rechte fügt ein.



57. Zusätzlich zu dem **kopierten Zeichen** wird auch noch ein anschließendes **c** benötigt. Zum bestätigen die **Strg**-Taste drücken.



 Wenn es geklappt hat, dann beendet er die aktuell laufende Anwendung und zeigt den Pfad zum aktuellen Verzeichnis an. Im Beispiel unten wurde der laufende RMFRecorder beendet.

Seite 48 von 50

과 Session A - [43 x 80]			_ 🗆 🛛
File Edit View Communication Actions Window ZipPrint Help			
0 E E # # # # # # & & & # @ @ @			
18 IBMzOS_OperatingSystem	LocalPI	8D1020#	scp Local P
erformance Index			
19 IBMzOS_OperatingSystem	SysplexPI	8D1020#:	#scp Sysplex
Performance Index	D-1		804750#
20 IBM20S_OperatingSystem	DelayPercentag	e	aDI (E0#scp
21 IBMZOS OperatingSustem	DelauForCPPerc	entage	8D3740#scp
Delays for CP percentage	being of offere	entage	000140#3CP
22 IBMzOS OperatingSystem	DelauForProces	sorPercen	tage 8D3790#
scp Delays for Processor percentage	2		
23 IBMzOS_OperatingSystem	TotalAAPonCPTi	mePercent	age 8D2D00#
scp Total zAAP on CP time percentag	3e		
24 IBMzOS_OperatingSystem	TotalIIPonCPTi	mePercent	age 8D35D0#
scp Iotal zIIP on CP time percentag			000750#
25 IBM208_Operating8ystem	DelayForHAPPer	centage	BD37E0#SCP
26 IBMZOS OperatingSustem	DelauForTIDDer	centage	8D3830#scp
Delay for zIIP percentage	betagi of fiffer	centage	000000#3CP
27 IBMzOS OperatingSystem	RGCappingDelay	Samples	8D3880#scp
Resource group capping delay samples p	ercentage		
28 IBMzOS_OperatingSystem	SRBTimePercent	age	8D2D40#scp
SRB time percentage			
29 IBMzOS_OperatingSystem	TCBTimePercent	age	8D2D50#scp
ICB time			
I. query finished Metaica incented in DB-(ÁČÉŇÖČ/%ÍÓ			
duration 15 sec	-aeele-H-/NHa	d	
sleep 17 sec			
Signal SIGINT caught			
Shutdown Program			
Shutdown RMFReader			
Connection to CIM-Server closed.			
Shutdown Database			
RMFReader balted			
WIM4:PRAK1:/u/prak1/v4/dist>			
a contract of the second sec			
= = =			
			RUNNING
ESC=Ý 1=Help 2=SubCmd 3=HlpRe	trn 4=Top	5=Bottom	6=TSO
7=BackScr 8=Scroll 9=NextS	ess 10=Refresh	11=FwdRet	r 12=Retrieve
MA	Û		40/007
Connected to remote server/host tn3270.de.ibm.com using lu/pool FU0U5224 and port 23			11

59. Praktischerweise besitzt der hier verwendete Terminal-Emulator eine **History-Funktion**. Mit den Tasten **F11** und **F12** kann durch die schon verwendeten Befehle gescrollt werden. Somit muss man den befehl nicht jedes mal aufs Neue zusammenbauen

Seite 49 von 50

7 Sonstige wichtige Hinweise und Tips

Dieses Kapitel behandelt sonstige Tips, Tricks und andere Hinweise, die die Bedienung erleichtern oder Helfen Fehler zu vermeiden. Momentan ist keine Struktur vorhanden, es ist einfach nur eine Liste die möglichst kurz die Fakten auf den Punkt bringt.

- Der ABLE-Editor ist in Eclipse das Projekt ablegui
- Die Klasse AbleDBImportBeanInfo.java im WLPrediction-Projekt wurde von Hagmann aus Projekt ablegui importiert, nicht aus able!
- WICHTIG: Bei der IBM ist es üblich, dass alle paar Monate sämtliche Passwörter geändert werden müssen. Das kann zu ernsthaften Problemen führen. Es muss garantiert werden, dass sämtliche Vorkommen der Benutzername/Passwort-Kombo mit dem neuen Passwort ersetzt werden!

Wenn man das Passwort mehrmals falsch eingibt, dann wird der Benutzer für den Großrechner gesperrt. Da zum Verbinden mit der Datenbank das selbe Passwort verwendet wird, kann es schnell passieren, dass man sich indirekt über die Anwendung mehrmals mit dem falschen Passwort anmeldet und in der Folge der Accout gesperrt wird! Am besten beim Ersetzen des Passworts die Suchfunktion von Eclipse verwenden, damit nichts vergessen wird (3x im Quellcode, 3x in WLPDefault.properties).

- Nach einer **Passwortänderung funktionieren** die **WlpAgenten** auch **nicht mehr**. Eine Änderung ist meines Wissens nicht möglich, die Agenten müssen neu erstellt werden.
- WICHTIG: Zum Starten des ABLE-Editors (ablegui), müssen folgende Argumente als VM arguments in der Run configuration mitgegeben werden:
 -Dable.home="\${workspace_loc:WLPrediction\Able_2.5.0}" -Dable.prefdir="\$ {workspace_loc:WLPrediction}"
- WICHTIG: Im ABLE-Editor muss unter Preferences... das Plugin Jar Directory spezifiziert werden. Das ist das Verzeichnis für die selbst erstellten ABLE-Beans. Ohne diese Angabe ist es im ABLE-Editor nicht möglich einen WlpAgent zu erstellen! Im vorliegenden Fall wäre das:

...\Diplomarbeit\workspace\WLPrediction\Able_2.5.0\lib\plugin

- Nach jeder erfolgten Änderung im WLPrediction-Projekt, sollte man das Projekt neu Erstellen, um vom ABLE-Editor verwendete jar-Dateien (wlpcore.jar,...) auf den neuesten Stand zu bringen.
- Für die graphische Anzeige der Vorhersage folgende URL verwenden: <u>http://localhost:8081/img/basicChart?table=prediction</u> Außer dem Parameter 'table' gibt es auch noch 'begin', 'end' (beides als Timestamp), so wie 'length' für die Anzahl anzuzeigender Datensätze. Note: Außer 'table' hat bei mir nichts geklappt.
- Die Daten f
 ür neu erstellte WlpAgents wurden hart reingecoded (Metricld, Instanceld, Timeintervall,...). Das wird in der Klasse WlpAgentenImpl.java erledigt.
- Um das vorhanden WLP-Framework das erste mal in Eclipse einzubinden, kann man bei der Frage nach dem Workspace einfach den Pfad zum alten Workspace angeben.

Seite 50 von 50

B. Inhalt der beigefügten DVD

Die beigefügte DVD weist folgenden Inhalt auf:

- Verwendete *Eclipse*-Software mit funktionsfähigem *WLP-Framework* im *Workspace*
- Generell verwendete Literatur und andere Diplomarbeiten im Rahmen des Joint Research Projects
- JDBC-Treiber für das Herstellen einer Verbindung zur *DB2 for z/OS* Datenbank
- Lauffähige Version des *SQuirreL* SQL-Clients (*Weekly Snapshot*) inklusive verwendeter SQL-Skripte
- Sonstige zur Verwendung heruntergeladene Software

Die komplette Verzeichnisstruktur findet sich zusätzlich in Abbildung B.1.



Abbildung B.1.: Inhalt der beigefügten DVD

Literaturverzeichnis

- [Apa08] Apache DB Project. Apache Derby. Webseite, Dezember 2008. http: //db.apache.org/derby.
- [Cod08] Codehaus. What is jaxen? Webseite, Dezember 2008. http://jaxen. codehaus.org/faq.html.
- [Dis08a] Distributed Management Task Force (DMTF). Common Information Model (CIM) Standards. Webseite, Dezember 2008. http://www.dmtf.org/ standards/cim/.
- [Dis08b] Distributed Management Task Force (DMTF). WBEM: CIM-XML protocol specification. Webseite, Dezember 2008. http://www.dmtf.org/ standards/wbem/CIM-XML.
- [Geb06] Clemens Gebhard. Datengewinnung und Parameterbestimmung zur Lastvorhersage in z/OS. Master's thesis, Eberhard Karls Universität Tübingen, October 2006.
- [Hag07] Michael Hagmann. Architektur und Integration der Workload-Vorhersage auf Basis neuronaler Netze in z/OS. Master's thesis, Eberhard Karls Universität Tübingen, October 2007.
- [Har08] Elliotte Rusty Harold. XOM FAQ: "What's the difference between XOM and JDOM?". Webseite, Dezember 2008. http://xom.nu/faq.xhtml# d0e24.
- [IBM06] IBM Press Release. IBM Transforms Database Market With Introduction of DB2. Webseite, Juni 2006. http://www-03.ibm.com/press/us/en/ pressrelease/19781.wss.
- [IBM07] IBM. DB2 9 pureXML Guide. Redbook sg247315, Januar 2007. http: //www.redbooks.ibm.com/abstracts/sg247315.html.

- [IBM08a] IBM. Agent Building and Learning Environment (ABLE). Webseite, Dezember 2008. http://www.alphaworks.ibm.com/tech/able.
- [IBM08b] IBM. DB2 Homepage. Webseite, Dezember 2008. http://www-01.ibm. com/software/data/db2/.
- [IBM08c] IBM. IBM Data Studio Developer. Webseite, Dezember 2008. http://www-142.ibm.com/software/dre/ecatalog/detail.wss?locale= de_DE&synkey=R491071Z63067U48.
- [IBM08d] IBM. z/OS Capacity Provisioning. Webseite, Dezember 2008. http://www-03.ibm.com/servers/eserver/zseries/zos/wlm/release_ history/cp/index.html.
- [JDO08] JDOM Project. JDOM Project Mission. Webseite, Dezember 2008. http: //www.jdom.org/mission/index.html.
- [Kle06] Sarah Dorothea Kleeberg. Neuronale Netze und Maschinelles Lernen zur Lastvorhersage in z/OS. Master's thesis, Eberhard Karls Universität Tübingen, August 2006.
- [Kry08] Krysalis Community Project. JCharts. Webseite, Dezember 2008. http: //jcharts.sourceforge.net.
- [Met08] MetaStuff, Ltd. dom4j FAQ: "How does dom4j relate to JDOM?". Webseite, Dezember 2008. http://www.dom4j.org/dom4j-1.6.1/faq.html# whats-dom4j-v-jdom.
- [Mor08] Mort Bay Consulting. Jetty. Webseite, Dezember 2008. http://www. mortbay.org/jetty.
- [Ope08] Open Group. OpenPegasus: "C++ CIM/WBEM Manageability Services Broker". Webseite, Dezember 2008. http://www.openpegasus.org.
- [Seu08] Holger Seubert. Umsetzen einer Persistenzstrategie mit Data Studio Developer & pureQuery. PDF, Dezember 2008. http://www-05.ibm.com/ de/events/data-studio/pdf/Data-Studio-Developer-1_2.pdf.
- [Sta08] Standards Based Linux Instrumentation for Manageability (SBLIM). CIM Client for Java. Webseite, Dezember 2008. http://sblim.wiki. sourceforge.net/CimClient.

- [Ull08a] Christian Ullenboom. Galileo Computing: Java ist auch eine Insel. Webseite, Dezember 2008. http://openbook.galileocomputing.de/ javainsel7.
- [Ull08b] Christian Ullenboom. Galileo Computing: Java ist auch eine Insel, Kapitel 14.3: Die Java-APIs für XML. Webseite, Dezember 2008. http://openbook.galileocomputing.de/javainsel7/javainsel_14_ 003.htm#mjd62c1928c73c88bad15dc7a76a8c53bc.
- [Wan06] Jim Wankowski. Quest Software: Breaking DB2 Platform Barriers. Webseite, Oktober 2006. http://www.quest.com/documents/landing.aspx? id=1079&technology=2&prod=&prodfamily=&loc=.
- [WG08] Gerd Wagner and Glenn Griffin. SQuirreL, ein universeller SQLClient. PDF, Dezember 2008. http://www.squirrelsql.org/paper/SQuirreL_de. pdf.
- [Wik08] Wikipedia. DB2 History. Webseite, Dezember 2008. http://en.wikipedia. org/wiki/IBM_DB2#History.
- [Wor08a] World Wide Web Consortium (W3C). Document Object Model (DOM). Webseite, Dezember 2008. http://www.w3.org/DOM.
- [Wor08b] World Wide Web Consortium (W3C). DOM-Baum für Beispiel-Tabelle. Webseite, Dezember 2008. http://www.w3.org/TR/2004/ REC-DOM-Level-3-Core-20040407/introduction.html#ID-E7C30821.
- [Wor08c] World Wide Web Consortium (W3C). XML Path Language (XPath). Webseite, Dezember 2008. http://www.w3.org/TR/xpath.