


# DAS PRIMEBASE SYSTEM
# EIN ÜBERBLICK

*Juni 2001*
*SNAP Innovation Softwareentwicklunggesellschaft mbH*
*Altonaer Poststraße 9a - D – 22767 Hamburg - Germany*
*www.primebase.com - e-mail: info@primebase.com*

1. Einführung ..... 1

2. Die PrimeBase Komponenten ..... 2
Die PrimeBase Virtual Machine ..... 2
Der PrimeBase SQL Database Server ..... 3
Der PrimeBase Application Server ..... 3
Der PrimeBase Open Server ..... 4
Der PrimeBase Automation Client ..... 4

3. Der Aufbau der PrimeBase Virtual Machine ..... 5
Datenfluss ..... 5
Komponenten-Hosting ..... 6
Sessions ..... 7
Erstellen und Beenden von Sessions ..... 7
Verbindungen ..... 8

4. Das PrimeBase Environment ..... 9
Lokale und globale Location ..... 9
Der PrimeBase System-ordner ..... 9
Systemdateien und Parameter ..... 11
Shared Memory und Komponentennamen ..... 11
Die PrimeBase Console ..... 12
Konfigurationen ..... 12

5. Schlussfolgerung ..... 13

# DAS PRIMEBASE SYSTEM - EIN ÜBERBLICK

PrimeBase ist ein integriertes, offenes Entwicklungssystem für Internet-, Intranet- und mobile Anwendungen (webbasierte Anwendungen). Diese Anwendungen basieren auf Internetstandards und haben zumeist ein Interface, das über einen Web-Browser gesteuert wird. PrimeBase ist ein integriertes System, da es alle notwendigen Komponenten zur Entwicklung solcher Anwendungen umfasst. PrimeBase ist ein offenes System, da jede Einzelkomponente in Verbindung mit der PrimeBase Virtual Machine eine Einheit bildet, die unabhängig eingesetzt und zusammen mit 3rd-Party-Produkten genutzt werden kann. Zu diesem Zweck sind in PrimeBase eine Vielzahl von Interfaces und Standards - wie Java, SQL, HTTP, XML, ODBC, JDBC, DAM, EOF u. a. - implementiert. Die zentralen Komponenten von PrimeBase sind in Fig. 1 dargestellt.

## 1. Einführung

Dieser Überblick beschreibt das komplette PrimeBase System und bereitet somit auf alle weitergehenden PrimeBase Dokumentationen vor. Die verschiedenen PrimeBase Komponenten arbeiten zusammen und bilden eine Komplettlösung. Darüber hinaus sind alle Einzelkomponenten mit der PrimeBase Virtual Machine verbunden (siehe das Kapitel “Der Aufbau der PrimeBase Virtual Machine” ). Es ist daher sinnvoll, wenn Anwender, Systemadministratoren und Programmierer von PrimeBase ein Verständnis der Funktionsweise jeder PrimeBase Komponente haben. Ein Überblick über das PrimeBase System ermöglicht die Beurteilung der Möglichkeiten, die PrimeBase zur Erfüllung von IT-Bedürfnissen bereitstellt.

Wie sich zeigen wird, gehen diese Möglichkeiten weit über die bloße Entwicklung von Webanwendungen hinaus. PrimeBase kann beispielsweise als Middleware zur Verbindung unterschiedlicher Systemlösungen eingesetzt werden. Außerdem kann PrimeBase als e-Business-Plattform dienen oder als Basis eines industriellen Controller-Systems fungieren.

Zusätzlich bieten wir eine Vielzahl von Anwendungslösungen, die auf der PrimeBase Technologie basieren - wie PrimeBase Job Market, PrimeBase Auction und PrimeBase E-Commerce. Weitere Lösungen vertreiben wir zusammen mit unseren Partnern. PrimeBase Anwendungslösungen sind komplett einsatzfähig, können aber auch als Basis für Ihre Weiterentwicklungen genutzt werden. Aufgrund der integriert-offenen Konzeption des PrimeBase Systems können PrimeBase Lösungen mit entsprechenden Erweiterungen als schlüsselfertige Systeme ausgeliefert werden.

Neben Geschwindigkeit, Stabilität und Zuverlässigkeit, die von einem System wie PrimeBase ohnehin erwartet werden, ist Einfachheit unser zentrales Anliegen. Auch wenn Einfachheit gegenwärtig leider allzu oft mit funktioneller Armut verwechselt wird, sind wir der Meinung, dass die Systeme der Zukunft, so wie PrimeBase, beides bieten müssen: Einfachheit und funktionelle Vielfalt.

Die Einfachheit des PrimeBase Systems bedeutet für Sie:

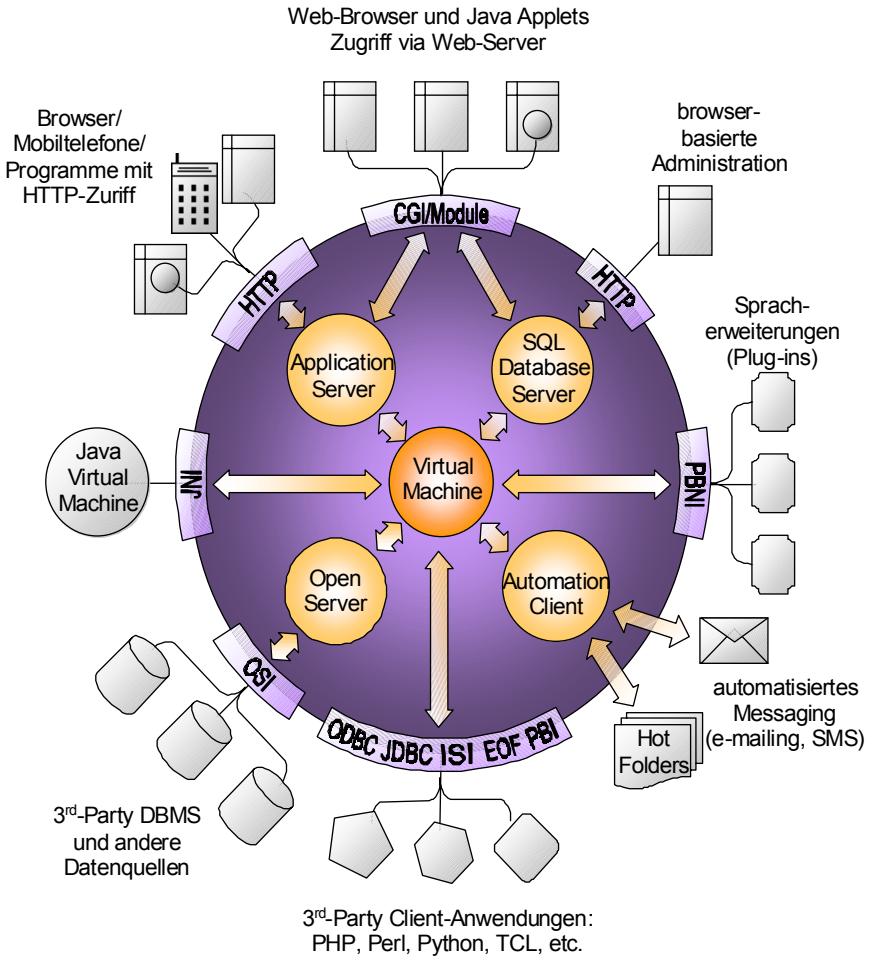
- einfache, schnelle und kostengünstige Installation durch "copy-and-start",
- geringe Wartungskosten, da alle PrimeBase Komponenten darauf ausgerichtet sind, ohne permanentes Monitoring zu arbeiten,
- minimaler Schulungsaufwand – da PrimeBase einfach zu verstehen ist, sind die Anwender schneller produktiv,
- vergleichsweise moderate Hardwareanforderungen.

Gegenwärtig stehen wir in der Welt der Computertechnologie einer Unmenge an Auswahlmöglichkeiten gegenüber. Für die Lösung der gleichen Aufgabe werden eine Vielzahl von unterschiedlichen Systemen und Standards angeboten. Wie soll da das entscheidende Auswahlkriterium aussehen? Wir glauben, dass, wenn man schon eine Auswahl hat, man die einfachste Lösung wählen sollte - nicht zuletzt,

weil das zu entwickelnde System wahrscheinlich ohnehin komplex genug ist. Dieser Maxime folgen wir bei der Auswahl derjenigen Standards, die in das PrimeBase System eingebunden sind.

## 2. Die PrimeBase Komponenten

Fig. 1 veranschaulicht die unterschiedlichen Komponenten des PrimeBase Systems sowie den Datenfluss zwischen diesen Komponenten.




*Figur 1: PrimeBase – ein integriertes, offenes System.*

Die diversen Interfaces, die die Integration von PrimeBase in bestehende Systeme und 3rd-Party-Lösungen ermöglichen, sind im Kreis rund um das PrimeBase System abgebildet. Im Zentrum des PrimeBase Systems steht - wie dargestellt - die PrimeBase Virtual Machine. Um diesen Kern finden sich in einem Kreis die vier anderen Hauptkomponenten des PrimeBase Systems: der PrimeBase SQL Database Server, der PrimeBase Application Server, der PrimeBase Open Server und der PrimeBase Automation Client.

### Die PrimeBase Virtual Machine

Die PrimeBase Virtual Machine (PBVM) ist die zentrale Verarbeitungseinheit des PrimeBase Systems und somit von hoher Bedeutung für das ganze System. Die PBVM ist für den Datentransfer und die Datenmanipulation ebenso verantwortlich wie für die Einbindung externer Arbeitsvorgänge in das PrimeBase System. Das umfasst die nahtlose Anbindung an die Java Virtual Machine (mittels Java Native Interface) und die Einbindung von Spracherweiterungen, die weitere Funktionalitäten über "plug-ins" bereitstellen (mittels PrimeBase Native Interface).

Die PrimeBase Virtual Machine ist programmierbar und unterstützt alle gebräuchli-

chen Konstrukte und Anweisungen sowie die Erstellung von Klassen und Objekten in einer mit Java vergleichbaren Weise. Die PrimeBaseTalk (PBT) benannte Programmiersprache unterstützt außerdem den gesamten SQL-Standard (Structured Query Language) für den Datenbankzugriff. PBT kennt alle Datenbank-Datentypen, auch NUMERIC, UNICODE, BLOBs und NULL, wodurch sich die PBVM ideal für den Austausch, die Umwandlung und die Filterung von Geschäfts- und Mediendaten eignet.

Aufgrund der zentralen Bedeutung der PBVM für das PrimeBase System wird die Funktionsweise dieser Komponente in einem eigenen Kapitel “Der Aufbau der PrimeBase Virtual Machine” weiter unten detaillierter beschrieben.

## Der PrimeBase SQL Database Server

Der PrimeBase SQL Database Server (PBDS) ist ein relationales Datenbanksystem. Neben dem ANSI SQL'92-Standard unterstützt PBDS Unicode, Binary und Character Large Objects und Volltextindizierung. Dadurch ist PBDS ideal für die Entwicklung von dynamischen Internet- und Intranet-Websites geeignet.

Außerdem lässt sich PDBS - wie in Fig. 1 gezeigt - webbasiert administrieren. Datenbanken können so über eine einfach zu bedienende Browsermaske erstellt, verändert und administriert werden. Dieses Interface wird PBDS Admin Server genannt.

Auf klassischer Client/Server-Technologie basierend, kann PBDS als ein eigenständiges DBMS (Datenbank Management System) genutzt werden. Dabei wird der Zugang zu PBDS über die PrimeBase Virtual Machine hergestellt, die alle gängigen Zugriffsstandards für Datenbanken unterstützt:

- ODBC: Microsofts Open Database Connectivity
- JDBC: Sun Microsystems Java Database Connectivity
- ISI: das PrimeBase Item Stream Interface, das auf Apples Data Access Manager (DAM/DAL) basiert
- EOF-Adapter: Apples Enterprise-Object-Framework-Adapter
- PBI: das PrimeBase Interface, das alle Funktionen von ODBC bei geringerer Komplexität umfasst

So gesehen dient die PrimeBase Virtual Machine dem PrimeBase SQL Database Server als Client-Software. Wie wir gesehen haben, ist diese Tätigkeit als Client-Software nur ein Aspekt der Arbeit der PrimeBase Virtual Machine, wenn auch ein wichtiger.

Das bedeutet beispielsweise, dass, wenn man den PrimeBase ODBC-Treiber installiert, um eine traditionelle Client/Server-Anwendung wie MS Access oder Delphi laufen zu lassen, die PrimeBase Virtual Machine ebenfalls auf dem Rechner installiert wird. Der ODBC-Treiber gibt die Daten an die PBVM weiter, die wiederum die SQL-Anweisungen kompiliert und an den PrimeBase SQL Database Server oder den PrimeBase Opern Server sendet.

## Der PrimeBase Application Server

Eine weitere zentrale Komponente des PrimeBase Systems ist der PrimeBase Application Server (PBAS), der so genannt wird, weil er Anwendungen auf Anforderung von Client-Programmen ausführt. Im Fall des PrimeBase Application Servers ist das entweder ein Webbrowser oder irgendein anderer Internet- oder Intranet-User-Agent wie ein Java-Applet oder sogar ein anderer Application Server.

„Application Serving" fügt dem herkömmlichen Client/Server-Modell eine dritte Ebene hinzu. Aus diesem Grund stellen Application Server eine dreistufige Architektur bereit. Diese drei Stufen sind die Benutzerschnittstelle (Client-Programm), die Anwendung (Arbeitslogik) und die Datenbank (Informationssicherung).

Diese Architektur erfreut sich mittlerweile - insbesondere seit dem Aufkommen des Internets - aufgrund seiner Vorteile großer Popularität, da sie z. B. die Administration, die Wartung und die Zugriffskontrolle von Anwendungen zentralisiert.

Der PrimeBase Application Server unterstützt den Internet-Standard für Datentransfer, HTTP, und arbeitet zudem mit Webservern zusammen, um eine Verbindung zwischen dem User Agent im Internet und dem PrimeBase System herzustellen. Dafür wird eine PrimeBase Technologie namens "LiveTags" benutzt. LiveTags™ sind eine Technik, bei der HTML- oder XML-Tags mit dynamischem Inhalt gefüllt bzw. durch diesen ersetzt werden. Dadurch können, was die Administration einer Anwendung erheblich erleichtert, Code und Layout einer jeden Anwendung vollständig voneinander getrennt werden.

Aufgrund seiner Unterstützung des Common Gateway Interface (CGI) und von Native Modules und Shared Libraries läuft PBAS mit einer Vielzahl von Webservern wie Apache, MS Internet Information Server, iPlanet Webserver (früher: Netscape) und WebSTAR.

Um eine Anwendung ausführen zu können, ist der PrimeBase Application Server auf die PrimeBase Virtual Machine angewiesen. Damit erhält die jeweilige Anwendung mittels der PrimeBase Virtual Machine vollen Zugriff auf das gesamte PrimeBase System und etwaige 3rd-Party-Produkte.

Weitere Details über den PrimeBase Application Server vermittelt der "PrimeBase Application Server User's Guide".

## Der PrimeBase Open Server

Der PrimeBase Open Server (PBOS) bietet Zugriff auf externe Datenbank Management Systeme (DBMS) oder andere Datenquellen. Erreicht wird dies durch die Entwicklung von PrimeBase Open Server Inferface (PBOSI) kompatible Plug-ins.

Diese Plug-ins sind für die Übersetzung der Daten und SQL-Anweisungen in die DBMS-spezifische Form zuständig, wie sie von 3rd-Party-Systemen gefordert wird. Somit können mit dem PrimeBase System Anwendungen realisiert werden, die vollkommen DBMS-unabhängig sind.

Der PrimeBase Open Server kann als eigenständige Anwendung installiert werden, aber er ist ebenfalls Teil der PrimeBase Virtual Machine. Das bedeutet, dass sowohl die PBVM als auch der PrimeBase Open Server als eigenständige Anwendung jeweils ein PBOSI Plug-in laden können.

Welche Konfiguration man wählt, hängt letztendlich vom System ab. Für maximale Effizienz empfiehlt sich der direkte Zugriff über die PBVM, was jedoch nur möglich ist, wenn ein PBOSI Plug-in des verwendeten DBMS für das entsprechende Betriebssystem erhältlich ist, auf dem die PBVM läuft. Ist das nicht der Fall, dann kann die Plattformunabhängigkeit des PrimeBase Systems genutzt werden, indem man PBOS einfach auf dem Betriebssystem installiert, auf dem schon das DBMS läuft und die PBVM für den Transport der Daten über das Netzwerk hin zum DBMS nutzt.

Während der Entwicklungsphase ist der eigenständige Betrieb einer PBOS-Anwendung sehr nützlich, da PBOS im "development mode" alle Aktivitäten zwischen der PBVM und einer 3rd-Party-Datenbank verfolgen kann. Dies hilft dem Entwickler bei der Suche nach ungültigen Statements und anderen Problemen.

## Der PrimeBase Automation Client

Der PrimeBase Automation Client (PBAC) ist der "Arbeiter" des PrimeBase Systems. PBAC ist dafür verantwortlich, regelmäßige Aufgaben und Batch-Jobs auszuführen und im Auftrag des übrigen Systems abzuarbeiten.

Wie in Fig. 1 gezeigt, umfasst das Aufgaben wie das Senden von E-Mails und sogenanntes "hot-foldering". Beim "hot-foldering" wird ein Ordner im Dateisystem vom Automation Client überwacht, um eintreffende Daten oder eintreffenden Code sofort vom Automation Client verarbeiten oder laden und ausführen zu lassen.

Es ist auch möglich, einen Timer im Automation Client zu setzen, um eine Aufgabe in regelmäßigen Intervallen ausführen zu lassen, was z. B. das regelmäßige Bakkup einer Datenbank umfassen kann.

Der PrimeBase Automation Client bedient sich - ebenso wie der PrimeBase Application Server - der PBVM zur Ausführung von anfallenden Aufgaben. Mit anderen

Worten bestimmt PBAC, wann eine Aufgabe gestartet und die entsprechenden Daten oder der Code verarbeitet werden sollen, überlässt dann aber der PBVM die Hauptarbeit.

In diesem Sinne verhält sich PBAC zur PrimeBase Virtual Machine wie eine Shell und hat auch - wie andere Shells (z. b. die UNIX command shell) - einen interaktiven Modus, der die Eingabe von Codefragmenten zur direkten Verarbeitung durch die PBVM erlaubt.

Das umfasst SQL- und alle anderen PBT-Anweisungen, die durch die PBVM unterstützt werden, sowie die Erstellung und den Gebrauch von PBT- und Java-Objekten. Daher können Programmierer den PBAC während der Entwicklungsphase einer Anwendung zum Test von SQL-Anweisungen und anderen Programmkomponenten nutzen.

Im interaktiven Modus können Daten aus jeder Datenquelle, die mit dem PrimeBase System verbunden ist, abgefragt und angezeigt werden. Dies ist möglich, da der PrimeBase Automation Client über die PBVM vollen Zugang zu allen anderen Komponenten des PrimeBase Systems hat.

# 3. Der Aufbau der PrimeBase Virtual Machine

Die PrimeBase Virtual Machine ist - als die zentrale Komponente des PrimeBase Systems - der Schlüssel zum Verständnis des gesamten Systems. Wie bereits angemerkt, sind alle anderen Komponenten des PrimeBase Systems auf die PBVM in der einen oder anderen Form angewiesen. In diesem Kapitel wird die Beziehung der PBVM zu den anderen Komponenten erklärt und es werden einige Details ihres Aufbaus deutlicher hervorgehoben.

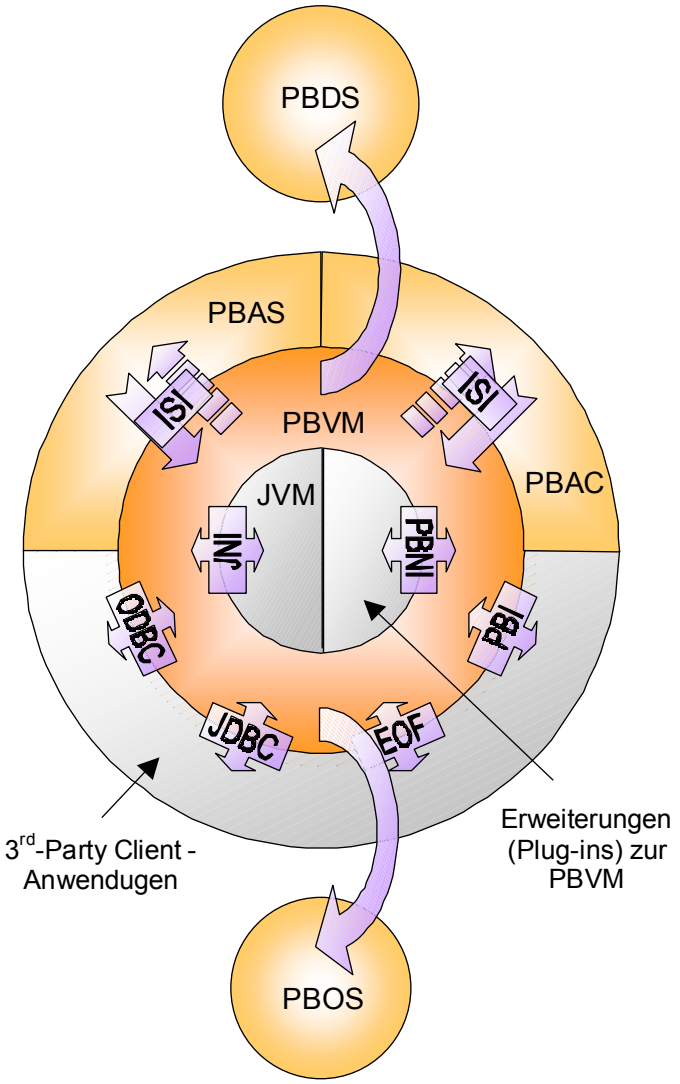
## Datenfluss

Wenn wir die PrimeBase Virtual Machine näher unter die Lupe nehmen, dann ist das erste, was wir bemerken, dass die Daten, die zu oder von anderen Komponenten des Systems fließen, sich in Abhängigkeit vom Typ der jeweiligen Komponente unterscheiden. Das wird in Fig. 2 anschaulich gezeigt.

Der Datenfluss zwischen den Komponenten kann in drei Gruppen eingeteilt werden:

- Shared Library Call Interfaces: ODBC, JDBC, EOF Adapter, PBI (PrimeBase Interface), JNI (Java Native Interface) und das PBNI (PrimeBase Native Interface) sind Standard Shared Library Call Interfaces. Daten werden direkt durch diese Aufrufe ausgetauscht. Diese Art von Interface wird von vielen 3rd-Party-Anwendungen genutzt.
- Item Stream Interfaces: Dieses Interface wird vom PrimeBase Application Server und dem PrimeBase Automation Client genutzt. Das "Item Stream" stellt die Standard-Output-Mechanismus der PBVM dar, in deren Datenstrom jedes Element einen Typ, eine Länge, einen Umfang (für fixed point values), Flags und Daten umfasst. Das PrimeBase Item Stream Interface kann auch für 3rd-Party-Anwendungen genutzt werden (in Fig. 2 nicht illustriert).
- Network Transport Interfaces: Wie in Fig. 2 zu sehen, tauscht die PBVM Daten mit dem PrimeBase SQL Database Server und dem PrimeBase Open Server über die gebräuchlichen Netzwerkprotokolle aus. Dies umfasst TCP/IP, AppleTalk und Shared Memory - letzteres dann, wenn die Virtual Machine und der Server auf dem gleichen Rechner laufen. Requests werden an den Server in Form von SQL-Anweisungen gesendet, woraufhin die Virtual Machine Seiten mit Daten als Antwort vom Server empfängt.

Im Hinblick auf das Network Transport Interface ist anzumerken, dass die PrimeBase Virtual Machine und der PrimeBase SQL Database Server zu einer “Laufzeit“-Version miteinander verbunden werden können. In diesem Fall ist dann das Interface eigentlich ein Call-Level-Interface und wird Internal/Runtime-Protokoll genannt. Darüber hinaus ist die PBVM immer mit einer Laufzeit-Version des PBOS verbunden. Daher kann das Internal/Runtime-Protokoll auch für den Zugriff auf den PrimeBase Open Server genutzt werden.




*Figur 2: Datenfluss von und zur PBVM*

## Komponenten-Hosting

Ein anderer in Fig. 2 dargestellter Aspekt der PrimeBase Virtual Machine ist das Konzept des "Komponenten-Hostings". Genauer gesagt wird die PBVM durch die Anwendungen, die auf sie mittels Call-Level-Interface, ODBC, JDBC, EOF, PBI oder ISI zugreifen, "gehostet". Die PBVM wiederum hostet die Java Virtual Machine oder andere systemeigene Erweitrungen, auf die über JNI oder PBNI zugegriffen wird.

Die hostenden Komponenten sind in Fig. 2 jeweils um die gehosteten Komponenten angeordnet. In der Praxis ist die gehostete Komponente immer eine Dynamic Link Library (auch bekannt als Shared Object oder Code Fragment, abhängig vom Betriebssystem). Die weiter außen stehende Host-Komponente muss immer eine Anwendung sein. Dabei handelt es sich in der Regel - wie Fig. 2 zu entnehmen - um den PrimeBase Application Server, den PrimeBase Automation Client oder eine 3rd-Party-Anwendung.

Die gehosteten Komponenten bilden eine sogenannte Laufzeitumgebung, die einen eigenen Datenbereich hat und unabhängig von der hostenden Komponente läuft. Die Synchronisation des Hosts und der gehosteten Komponente geschieht über das gemeinsame Interface.

## Sessions

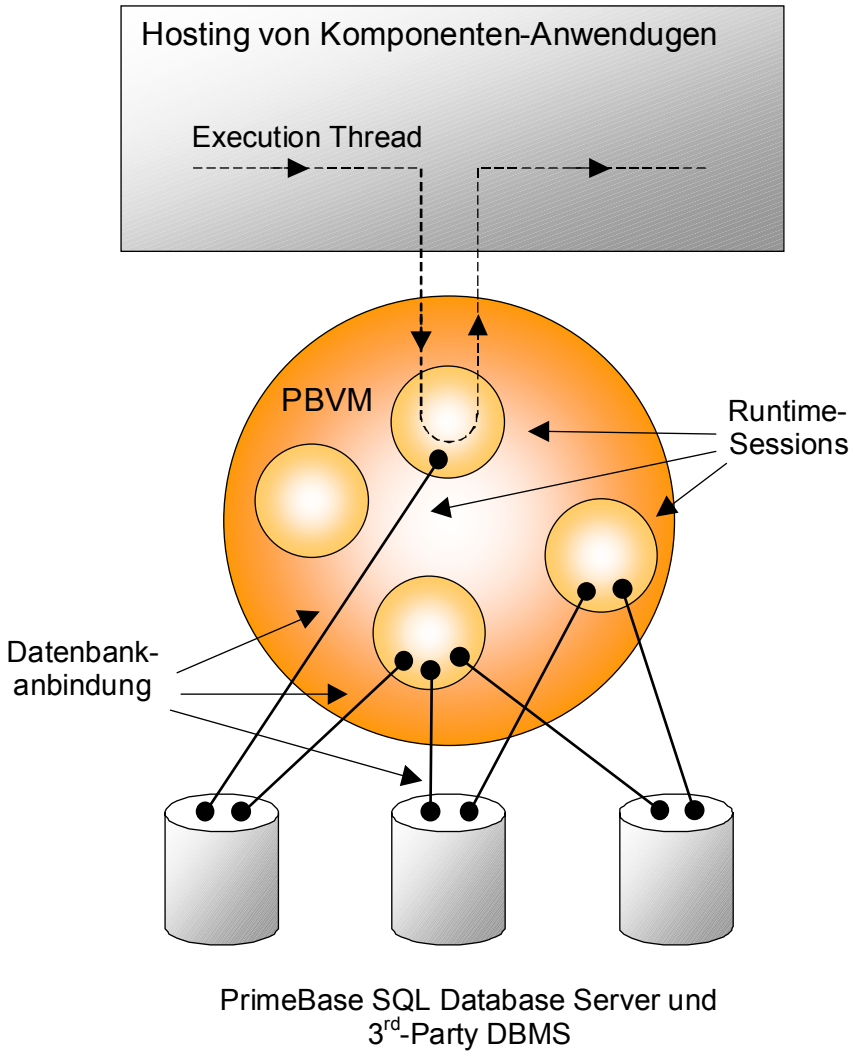
Das Session-Konzept ist, wie bei einem genaueren Blick auf den Arbeitsablauf der PrimeBase Virtual Machine deutlicher wird, ein wichtiger Aspekt des Aufbaus derselben. Fig. 3 veranschaulicht den internen Aufbau der PBVM.

Eine Session (Sitzung) hat ihren eigenen Speicherbereich innerhalb der Laufzeitumgebung der PrimeBase Virtual Machine. Jede Sitzung hat einen Execution Thread, der durch die hostende Komponente veranlasst wird. Eine Session kann auch als Anwendung oder "Programminstanz" bezeichnet werden.

Die Virtual Machine kann mehrere Instanzen des gleichen PBT-Programms jeweils in einer eigenen Session starten. Die Sessiongrenzen gewährleisten, dass die einzelnen Instanzen unabhängig voneinander laufen. Programminstanzen können sich gegenseitig nicht beeinflussen, da jede Session ihren eigenen Adressraum hat, wodurch sichergestellt ist, dass jede Programminstanz vor den Fehlern einer anderen Instanz geschützt ist.

## Erstellen und Beenden von Sessions

Für die Erstellung und Beendigung von Sessions stellt die PrimeBase Virtual Machine API (Application Programming Interface) level calls zur Verfühgung. Beim Erstellen einer Session lädt sie den notwendigen Code, um das Programm laufen zu lassen. Der geladene Code wird kompiliert, optimiert und lauffertig in einem plattformunabhängigen Bytecode gespeichert. Dies wird als die "Bootphase" einer Anwendung bezeichnet. Weiterer Code kann dann zur Ausführung oder Speicherung durch zusätzliche API level calls übermittelt werden.




*Figur 3: Interner Aufbau der PBVM*

Die Bootphase selbst ist in zwei Phasen, eine Initialisierungs- und eine Startup-Phase, unterteilt. In der Initialisierungsphase wird die "Clone-Session" des Systems zur Erstellung einer neuen Session dupliziert. In der Startup-Phase lädt die neue Session den Code, der in der Systemdatei 'startup.sys' spezifiziert wird (mehr über Systemdateien finden Sie in Kapitel “Das PrimeBase Environment” ).

Die Clone-Session wird von der Virtual Machine vor allen anderen Sessions durch das Laden des Codes aus der Datei 'initialize.sys' erstellt.

Beide Dateien, die 'startup.sys' und die 'initialize.sys', können weiteren Code und Daten laden, wie dies vom jeweiligen Programm erfordert wird, das von der PrimeBase Virtual Machine ausgeführt werden soll.

Da alle Sessions aus der Clone-Session erstellt werden, enthalten sie anfangs alle den Code und die Daten, die in der Datei 'initialize.sys' spezifiziert werden. Da die 'initialize.sys' gewöhnlich den gesamten Code lädt, den eine bestimmte Applikation benötigt, erlaubt der Cloning-Mechanismus die augenblickliche Einrichtung einer neuen Programminstanz.

Dabei handelt es sich um eine Basisanforderung von Webanwendungen. Wenn die PBVM für den Betrieb einer Internetanwendung benutzt wird, dann startet sie für jeden Nutzer, der sich mit dem System verbindet, eine neue Programminstanz. Genau in diesem Moment erstellt und "bootet" die Virtual Machine eine neue Session, was so schnell erledigt wird, dass der Nutzer keine wesentliche Verzögerung bis zum Eintreffen der ersten von ihm angeforderten Seite bei seinen Browser bemerkt.

Wenn eine Session beendet (geschlossen) wird, wird der in der Datei 'shutdown.sys' spezifizierte Code geladen und ausgeführt, was es der Programminstanz ermöglicht, Statusinformationen dauerhaft zu speichern.

## Verbindungen

In Fig. 3 wird auch die Verbindung zwischen der Laufzeitumgebung und den unterschiedlichen mit dem PrimeBase System verbundenen DBMS veranschaulicht. Das beinhaltet den PrimeBase SQL Database Server und andere 3rd-Party-DBMS, die über den PrimeBase Open Server angebunden werden können.

Es ist zu bedenken, dass jede Session keine, eine oder mehrere Verbindungen zu unterschiedlichen Datenbanken haben kann. Jede Verbindung wird dabei als "open DBMS" bezeichnet.

Wie viele Verbindungen jede Session benötigt, hängt vom Programm ab, dass von der Virtual Machine ausgeführt wird. Außerdem hängt es von der API ab, die von der hostenden Anwendung benutzt wird. So kennen z. B. APIs wie ODBC oder JDBC das Session-Konzept nicht. Diese APIs können nur Verbindungen verwalten. Im Ergebnis wird etwa der ODBC-Aufruf zum Aufbau einer Verbindung in einen Aufruf an die PrimeBase Virtual Machine übersetzt, die eine neue Session mit einer Verbindung einrichtet. Dadurch wird das Session-Konzept der PBVM für ODBC-Anwender transparent.

Wie bereits angemerkt, kann eine Session auch ohne jede Verbindung erstellt werden. Die Programminstanz, die in dieser Session läuft, wird dann auch einwandfrei funktionieren, es sei denn, sie benötigt den Zugriff auf Datenbankinformationen. Eine Session kann zu jedem beliebigen Zeitpunkt eine neue oder zusätzliche Verbindung mittels des von PBT bereitgestellten Befehls OPEN DBMS öffnen.

Eine Verbindungsdefinition (Connection Definition) enthält alle notwendigen Daten für eine Session, um eine neue Verbindung öffnen. Dies umfasst Informationen über das zu verwendende Übertragungsprotokoll (TCP/IP, AppleTalk, Shared Memory), den Hostnamen des Servers, den Servernamen und die Port-Nummer usw., die insgesamt Verbindungsoptionen (Connection Options) genannt werden. Verbindungsdefinitionen werden in der Systemdatei 'connect.def' gespeichert und haben jeweils einen eindeutigen Namen, den "Connection Alias".

# 4. Das PrimeBase Environment

Wenn Sie eine Komponente des PrimeBase Systems auf Ihrem Rechner installieren, dann wird diese zusammen mit einem Environment installiert, die aus mehreren Ordnern (Directories) mit System- und Steuerdateien besteht. PrimeBase Komponenten sind "vorinstalliert" in dem Sinne, dass zur Installation nur die ausführbaren Dateien (executables) zusammen mit den Ordnern und Dateien, die das Environment ausmachen, kopiert werden müssen - die PrimeBase Komponente kann dann umgehend gestartet werden.

In diesem Kapitel sollen einige Aspekte des PrimeBase Environment besprochen werden, die allen PrimeBase Komponenten gemeinsam ist. Ein Verständnis der dabei beschriebenen Konzepte wird Ihnen sowohl helfen, ein System, dass aus einer beliebigen Anzahl von PrimeBase Komponenten besteht, einzurichten und zu konfigurieren, als auch, ein bereits bestehendes System schnell zu verstehen.

## Lokale und globale Location

Für jedes PrimeBase Environment sind zwei Locations von besonderer Bedeutung – eine lokale und eine globale Location. Der Teil des PrimeBase Environments an der lokalen Location heißt "lokales Environment" und der Teil des PrimeBase Environments an der globalen Location wird "globales Environment" genannt.

Eine PrimeBase Komponente prüft, wenn sie eine Konfigurationsdatei oder einen Systemordner sucht, zunächst die lokale und dann erst die globale Location, was praktisch bedeutet, dass Konfigurationsdateien, die an der lokalen Location gespeichert werden, diejenigen Dateien an der globalen Location überschreiben.

Die tatsächliche Position dieser Locations im Dateisystem hängt von der Host-Plattform ab.

Lokale Location:

- auf Systemen unter UNIX/Linux oder Windows ist die lokale Location immer "das aktuelle Arbeitsverzeichnis".
- bei einem Macintosh ist die lokale Location der Ort der Anwendung.

Globale Location:

- bei UNIX/Linux ist die globale Location /usr/local/primebase. Die UNIX-Environmentvariable PRIMEBASE_HOME kann dafür genutzt werden, hierfür einen anderen Ordner anzugeben.
- unter Windows ist die globale Location der Windows-Systemordner.
- auf einem Macintosh-Rechner wird der "Preferences"-Ordner benutzt.

Wie bereits erwähnt, sind alle PrimeBase Komponenten vorinstalliert, was auch bedeutet, dass alle benötigten Control Files vorinstalliert an der lokalen Location liegen. In diesem Fall ist die lokale Location, die unter Windows und UNIX das aktuelle Arbeitsverzeichnis ist, die gleiche wie die Location der Komponente.

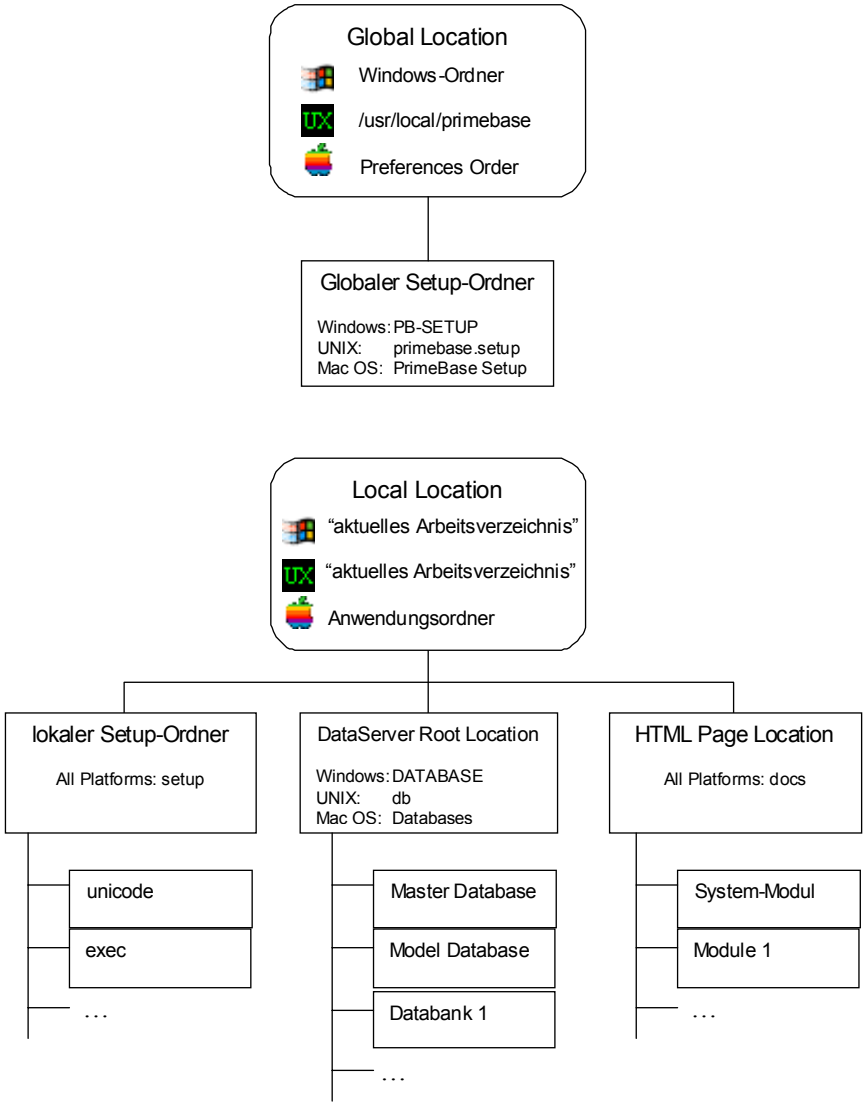
## Der PrimeBase Systemordner

An jeder Location kann man einen oder mehrere Systemordner finden. In Fig. 4 sind die zentralen Systemordner der PrimeBase Environment dargestellt.

Der Setup-Ordner des PrimeBase Systems enthält die System- und Kontrolldateien für alle PrimeBase Komponenten. Zusätzlich hierzu hat der PrimeBase Database Server einen Systemordner namens "DataServer Root Location" und der PrimeBase Application Server einen Systemordner namens "HTML Page Location".

Der Setup-Ordner des PrimeBase Systems heißt an der lokalen Location immer "Setup" - er wird auch "lokaler Setup-Ordner" genannt. An der globalen Location hängt der Name des Setup-Ordners ("globaler Setup-Ordner") vom Betriebsystem ab:

- auf UNIX/Linux-Systemen heißt der Ordner 'primebase.setup'
- unter Windows heißt er "PB-SETUP"
- bei einem Macintosh-Rechner wird er "PrimeBase Setup" genannt.




*Figur 4: die PrimeBase Systemordner*

Diese eindeutigen Namen stellen sicher, dass der globale Setup-Ordner auf jedem System leicht zu finden ist.

Die "HTML Page Location" und die "DataServer Root Location" befinden sich immer an der lokalen Location, da die an diesen Locations gespeicherten Dateien nicht von mehreren PrimeBase Komponenten geteilt werden können. Die "HTML Page Location" heißt auf allen Systemen 'docs'.

Der Name der "DataBase Root Location" hängt wieder von der Plattform ab:

- unter UNIX/Linux heißt die "DataServer Root Location" 'db'
- unter Windows heißt dieser Ordner 'DATABASE'
- auf einem Macintosh-Rechner wird er 'Databases' genannt

Natürlich können sich unterschiedliche PrimeBase Komponenten die gleiche lokale Location teilen, jedoch muss es sich um unterschiedliche Komponenten handeln. Beispielsweise können sich nicht zwei Instanzen des PrimeBase SQL Database Server die gleiche lokale Location teilen.

Der PrimeBase Setup-Ordner enthält einige Unterverzeichnisse. Das 'unicode'-Unterverzeichnis enthält die Unicode-Mappingdateien, die vom System verwendet

werden. Es handelt sich dabei um Standard-Mappingdateien, die auch unter 'http://www.unicode.org' erhältlich sind.

Der 'exec'-Unterverzeichnis ist die Location für ausführbare Dateien, die von der PrimeBase Virtual Machine benutzt werden. Dieser Ordner enthält den bereits erwähnten Bootcode mit den Dateien 'initialize.sys', 'startup.sys' und 'shutdown.sys'.

Der Ordner der "DataServer Root Location" ist die Location für Datenbanken, die vom PrimeBase SQL Database Server gemountet werden. Jeder weiter Unterordner in diesem Ordner enthält eine Datenbank. Zwei Datenbanken, "Master" und "Model", sind hier immer vorhanden.

Der Ordner der "HTML Page Location" enthält die Module und Dokumente, die über den PrimeBase Application Server aufgerufen werden können. Jeder weitere Unterordner dieses Ordners enthält jeweils ein Modul. Installiert wird ein Modul einfach, indem man es in diesen Ordner verschiebt. Das mitgelieferte System-Modul enthält ein Administrations-Interface für den PrimeBase Application Server.

## Systemdateien und Parameter

Die Systemdateien von PrimeBase speichern Parameter und andere Informationen, die die unterschiedlichsten Aspekte des PrimeBase Systems kontrollieren. Diese Dateien liegen im Setup-Ordner des PrimeBase Systems oder einem seiner Unterordner.

Jede Komponente des PrimeBase Systems hat eine Environmentdatei, die die Werte unterschiedlicher Systemparameter dieser Komponente speichert. Der Name der Environmentdatei korrespondiert mit dem Namen der entsprechenden PrimeBase Komponente und hat die Dateiendung '.env'. So heißt z. B. das Environment-File der PrimeBase Virtual Machine 'pbvm.env'.

Der PrimeBase SQL Database Server speichert eine Kopie seiner Systemparameter in der Master-Datenbank. Die Werte der Environmentdatei werden daher in diesem Fall nur zum Booten des Servers benutzt. Danach wird auf die Werte aus der Master-Datenbank zurückgegriffen.

Die Werte in den verschiedenen Environmentdateien werden im Binärformat gespeichert und können mit dem PrimeBase Environment Editor bearbeitet werden. Im Fall des PrimeBase SQL Database Servers müssen die Parameter direkt in der Master-Datenbank gesetzt werden.

Als Alternative zum Environment Editor bieten der PrimeBase SQL Database Server und der PrimeBase Application Server zum Ändern der Systemparameter jeweils ein webbasiertes Interface (der PBDS Admin Server und das PBAS Systemmodul).

Wie bereits erwähnt, werden die Verbindungseinstellungen des PrimeBase Systems in einer Datei namens 'connect.def' gespeichert. Diese Textdatei kann mit jedem Standardeditor bearbeitet werden, wobei jede Zeile in der Datei jeweils eine Verbindungsdefinition repräsentiert. Verbindungsdefinitionen können auch mit dem PrimeBase Automation Client, dem PrimeBase Admin Server und dem PrimeBase Systemmodul erstellt oder geändert werden.

Verbindungsdefinitionen werden von der PrimeBase Virtual Machine oder dem PBDS Admin Server genutzt, um sich mit dem PrimeBase SQL Database Server oder einem anderen DBMS - via PrimeBase Open Server - zu verbinden.

Weitere Informationen zu den Kontrolldateien der einzelnen Komponenten des PrimeBase Systems sind den jeweiligen Dokumentationen dieser Komponenten zu entnehmen.

## Shared Memory und Komponentennamen

Auf einer einzelnen Maschine wird von den PrimeBase Komponenten ein gemeinsamer Speicherbreich benutzt. Dieser Speicherbereich kann von den PrimeBase Komponenten auch anstelle des TCP/IP-Protokolls zur Kommunikation nutzten.

Jede PrimeBase Komponente, die im Hintergrund eine Anwendung laufen lässt, hat einen eindeutigen Namen und "publiziert" sich selbst mit diesem Namen im

gemeinsamen Speicherbereich.

Die verschiedenen PrimeBase Komponenten haben als default folgenden Namen:

- der PrimeBase SQL Database Server heißt "PrimeServer"
- der Name des PrimeBase Application Servers ist "PrimeBase"
- der PrimeBase Open Server hat den Namen "OpenServer"
- der PrimeBase Automation Client wird "AutoTask1" genannt

Da diese Komponentennamen eindeutig sein müssen, muss man den Default-Namen einer Komponente ändern, wenn man von ihr mehrere Instanzen auf einer Maschine laufen lassen will. Allerdings braucht der Automation Client nur dann einen eindeutigen Namen, wenn er im normalen Interaktiv-Modus läuft.

Der Komponentenname ist ein Standard-Systemparameter und kann wie die anderen Systemparameter auch in der Environmentdatei oder - im Fall des PBDS - der Master-Datenbank geändert werden.

## Die PrimeBase Console

Die PrimeBase Console ist ein textbasiertes Interface für die verschiedenen PrimeBase Komponenten. Abhängig von der Komponente können mit der Console administrative Aufgaben erledigt und Statusinformationen abgefragt werden. Außerdem können die einzelnen Komponenten so konfiguriert werden, dass sie über die Console Fehlermeldungen, Warnungen und andere Meldungen ausgeben. Diese Informationen können für das Debugging und die Fehlersuche genutzt werden.

Beim Starten der PrimeBase Console wird eine Liste der Komponenten angezeigt, die auf der betreffenden Maschine laufen, aus der dann eine ausgewählt werden kann, an die sich die Console anbindet.

Alle Befehle, die in die Console eingeben werden, beginnen mit einem #-Zeichen. Alle Komponenten unterstützen #help, wodurch eine Liste der unterstützten Befehle ausgegeben wird. Durch die Eingabe von #halt kann jede PrimeBase Komponente beendet werden.

Zusätzlich zu den #-Befehlen erlaubt die Console des PrimeBase SQL Database Servers ein Login und die interaktive Eingabe und das Starten von PBT-Programmen. Das erlaubt das Ausführen anspruchsvoller Aufgaben – wie ein Backup oder eine Datenbank-Reorganisation – direkt von der PBDS Console aus.

Mit STRG + R kann die Console jederzeit verlassen werden, wobei die Anwendung im Hintergrund weiter läuft.

## Konfigurationen

Die PrimeBase Komponenten können mit unterschiedlichen Konfigurationen ausgeliefert und eingesetzt werden, was eine hohes Maß an Flexibilität bei der Planung des eigenen Systems ermöglicht. Abhängig von den eigenen Systemanforderungen können die unterschiedlichen Komponenten als Shared Library oder als Einzelanwendung eingesetzt werden.

Wie bereits erwähnt, kann der PrimeBase SQL Database Server als Shared Library mit der PrimeBase Virtual Machine kombiniert und direkt an eine Anwendung angebunden werden. Dies wird auch die PBDS Laufzeit-Version genannt.

Die Laufzeit-Version des PrimeBase SQL Database Servers wird für den Einsatz eines sogenannten "kompakten Systems" genutzt, dass keinen Datenbank-Server als Einzelanwendung benötigt. Zum Beispiel kann der PrimeBase Application Server mit einer Laufzeit-Version des PBDS ausgeliefert werden. Um über das Web an Daten aus der Datenbank zu gelangen, braucht man dann nur den PrimeBase Application Server zu starten, da dieser bei der Laufzeit-Version direkten Zugriff auf die Daten in der Datenbank hat.

Dieses Szenario ist auch deswegen möglich, weil der PrimeBase Application Server direkten http-Zugriff unterstützt, so dass man auf die Daten auch zugreifen kann, ohne eigens einen Webserver zu starten. Bei dieser Konfiguration ist aller-

dings keine browserbasierte Administration möglich.

Ebenfalls bereits erwähnt wurde, dass auch der PrimeBase Open Server als Einzelanwendung oder als Teil der PrimeBase Virtual Machine erhältlich ist.

Alle PrimeBase Komponenten sind außerdem darauf ausgerichtet, auch auf Read-Only-Medien wie z. B. CD-Roms zu laufen, was bedeutet, dass es möglich ist, ein PrimeBase-basiertes System ohne jede Installation direkt von einer CD aus zu starten.

## 5. Schluss-folgerung

Das PrimeBase System bietet eine komplette Entwicklungslösung für web-basierte Anwendungen. Da sie eine Vielzahl an Industriestandards unterstützen, können die verschiedenen PrimeBase Komponenten eigenständig eingesetzt oder in ein bestehendes System integriert werden. Außerdem unterstützt PrimeBase eine große Zahl von Hardware-Plattformen und Betriebssystemen - wie Windows, UNIX/Linux, Sun Solaris, IBM AIX oder Mac OS.

PrimeBase bietet eine solide Basis mit hoher Performanz und breiter Funktionalität bei einfacher Bedienungslogik. Somit eignet sich PrimeBase ideal für die Entwicklung von e-Business- und anderen komplexen Internet-Anwendungen.