



# CARUSO

TECHNISCHE PRODUKTINFORMATION

## Inhalt

<b>1</b>	<b>Systemdefinition.....</b>	<b>2</b>
<b>2</b>	<b>Technische Details für den Betrieb von CARUSO .....</b>	<b>2</b>
2.1	Webserver .....	2
2.2	Java Version .....	2
2.3	Unterstützte Datenbankmanagementsysteme.....	2
2.4	Datensicherheit .....	3
2.5	Systemanforderungen.....	3
2.6	Betriebssystem des Webservers .....	3
2.7	Sicherungen .....	3
<b>3</b>	<b>Strategien für die Installation von CARUSO .....</b>	<b>4</b>
3.1	CARUSO ist direkt mit der Produktivdatenbank verbunden .....	4
3.2	CARUSO kommuniziert mit einer Backupdatenbank .....	5
3.3	Server-Aufbau für Load-Balancing / Failover .....	6

## 1 Systemdefinition

CARUSO ist das Web-Frontend für die Backoffice-Anwendung ORBIS, mit der alle organisatorischen und verwaltungstechnischen Arbeiten rund um den Betrieb von Weiterbildungseinrichtungen abgewickelt werden können. Als Web-Frontend sorgt CARUSO dafür, dass bestimmte Daten, die zuvor in ORBIS erfasst und bearbeitet wurden, im Inter- oder Intranet zur Verfügung stehen. Dies können zum einen Daten mit informativem Charakter sein, z.B. Veranstaltungsdaten mit Terminen, Dozenten, Fächern etc., Zum anderen stellt CARUSO auch interaktive Funktionalität zur Verfügung, indem es z.B. Anmeldungen abwickelt und direkt in das Backoffice-System ORBIS übergibt.

## 2 Technische Details für den Betrieb von CARUSO

CARUSO ist eine auf Java-Technologie basierende Web-Anwendung.

Für die Erzeugung der Webseiten werden JSP-Seiten (JSP = Java Server Pages) herangezogen, welche mittels Scriptlets und eigens für diesen Anwendungsfall entwickelte Java-Tag-Bibliotheken die Anzeigedaten zur Verfügung stellen.

### 2.1 Webserver

Zum Betrieb der CARUSO-Anwendung wird ein javafähiger Webserver benötigt. Wir empfehlen das Produkt Apache Tomcat ab der Version 5.5.x. Es sind keine besonderen Konfigurationseinstellungen notwendig.

Um von der Anwendung Mails verschicken zu können, muss der Webserver Zugriff auf einen Mailserver über SMTP haben.

### 2.2 Java Version

Die Java-Version, welche unterhalb des jeweiligen Webserver liegt, sollte die Version 1.5 aufweisen.

### 2.3 Unterstützte Datenbankmanagementsysteme

CARUSO unterstützt die vier Datenbankmanagementsysteme

- Oracle
- Microsoft SQL-Server
- Informix
- Gupta/Unify SqlBase

Wie unter Java üblich werden die Datenbanken mittels JDBC angesprochen. Da die Kommunikation von CARUSO mit der Datenbank in der Regel durch eine Firewall erfolgt, muss in Abhängigkeit der eingesetzten Datenbank ein Kommunikationsport in der Firewall freigeschaltet werden. Die Standardports sind

- Oracle: 1521
- Microsoft SQL-Server: 1433
- Informix: 1533
- Gupta/Unify SqlBase: 2155

In Abhängigkeit von der Installation können die genannten Ports variieren. Bitte sprechen Sie vor zuvor mit Ihrem Datenbankadministrator.

Ein Datenbankpool wird von Caruso selbst bereitgestellt und verwaltet. Es ist dafür keine Konfiguration am Webserver erforderlich.

### 2.4 Datensicherheit

CARUSO legt natürlich großen Wert auf den Schutz Ihrer Daten. Um diesen gewährleisten zu können, verfügt CARUSO über ein durchdachtes Benutzermanagement.

Um Hackerattacken vorzubeugen, werden alle Eingaben konsequent gefiltert, so dass Sql-Injection (Unterschieben von Sql-Befehlen in Textzeilen) und Crosssitescripting (Einbinden von Javascript-Elementen, damit sich die Seite anders „verhält“ als gewollt) verhindert werden.

Um die optimalen Sicherheitsvorkehrungen treffen zu können wurden im Kapitel 3 exemplarisch zwei Strategien aufgezeigt, welche verschiedenen Arten Sicherheitsbedürfnissen decken.

### 2.5 Systemanforderungen

Für den Betrieb von CARUSO empfehlen wir einen modernen Mehrkern-Prozessor-Server (z.B. Intel Core2 Duo) mit mindestens einem Gigabyte Hauptspeicher. Diese Angaben basieren auf einer mittleren Zugriffszahl, wobei die Anforderungen natürlich bei höherem Seitenverkehr ansteigen können, so dass evtl. mehrere Prozessoren und mehr Speicher notwendig werden.

Die Netzanbindung ist abhängig von dem zu erwartenden Zugriff auf die Webseite und kann daher an dieser Stelle nicht näher spezifiziert werden.

### 2.6 Betriebssystem des Webservers

Das Betriebssystem des Webservers kann im Ermessen des Betreibers liegen, da ein Java-Webserver auf nahezu allen Betriebssystemen verfügbar ist.

Allerdings hat sich herausgestellt, dass die Performance von Java-Anwendungen auf Unix bzw. Linux Betriebssystemen am vorteilhaftesten ist.

### 2.7 Sicherungen

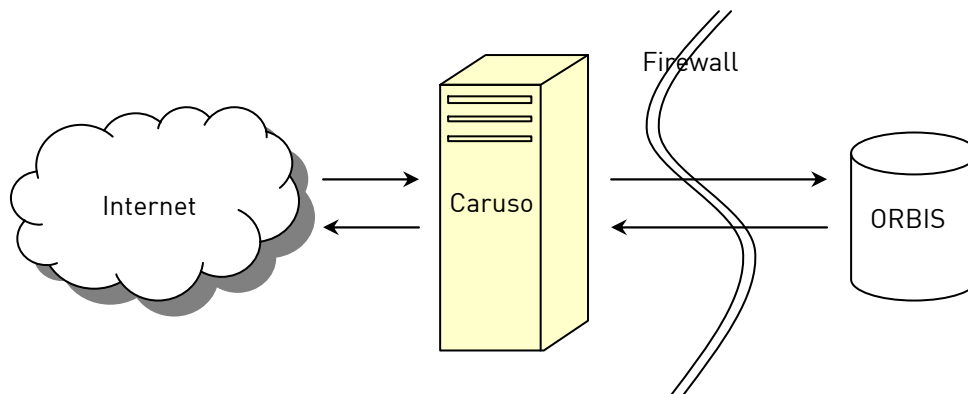
Die CARUSO-Installation (Webapplikation unter Tomcat) muss nur initial gesichert werden. CARUSO schreibt keine Anwendungsdaten in das Dateisystem, alle Daten werden in der Datenbank abgelegt. Für die Datenbank ist daher eine laufende Sicherung notwendig. Hierzu sollten die Tools des jeweiligen Datenbankherstellers genutzt werden.

## 3 Strategien für die Installation von CARUSO

Im Folgendem sollen zwei Strategien für den Betrieb von CARUSO sowie ein möglicher Serveraufbau dargestellt werden.

### 3.1 CARUSO ist direkt mit der Produktivdatenbank verbunden

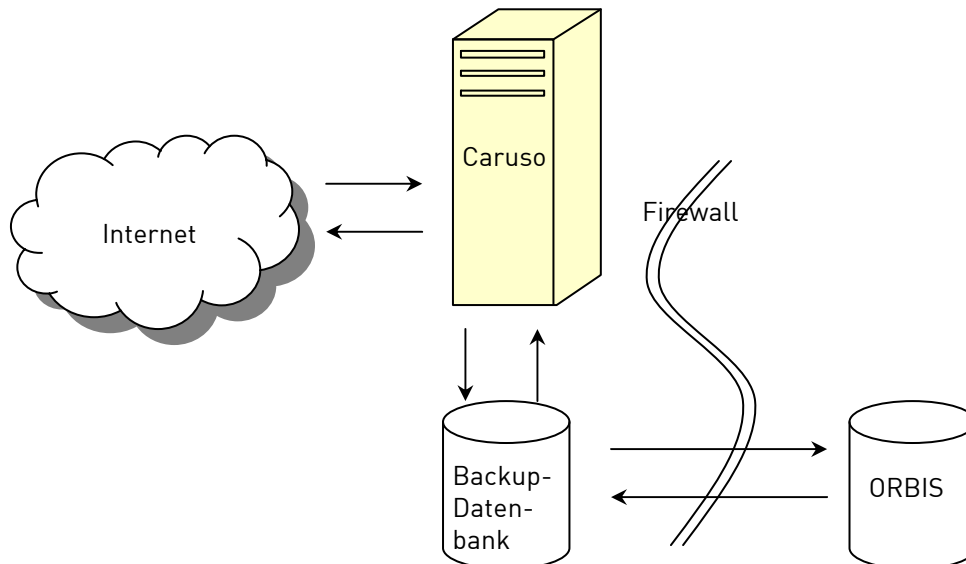
Die erste Strategie zeichnet sich durch den einfachen und übersichtlichen Aufbau aus.



Bei diesem Beispiel ist der Webserver, welcher CARUSO bereithält, direkt mit dem Internet verbunden. Um die Daten so aktuell wie möglich zu halten, greift CARUSO direkt auf die Produktivdatenbank von ORBIS zu. Diese Lösung hat den Vorteil, dass bei einer Neuinstallation von CARUSO die bestehende ORBIS-Architektur mitbenutzt werden kann. Hier ist keine zusätzliche Datenbank notwendig. Einzig der Webserver muss aufgebaut und administriert werden. Der Nachteil ist, dass bei einer eventuellen Sicherheitslücke im Webserver der direkte Zugriff durch die Firewall auf die Produktivdatenbank möglich ist. Durch die Benutzung eines nur für die CARUSO-Daten zuständigen Datenbankaccounts kann die mögliche Gefahr an dieser Stelle minimiert werden.

## 3.2 CARUSO kommuniziert mit einer Backupdatenbank

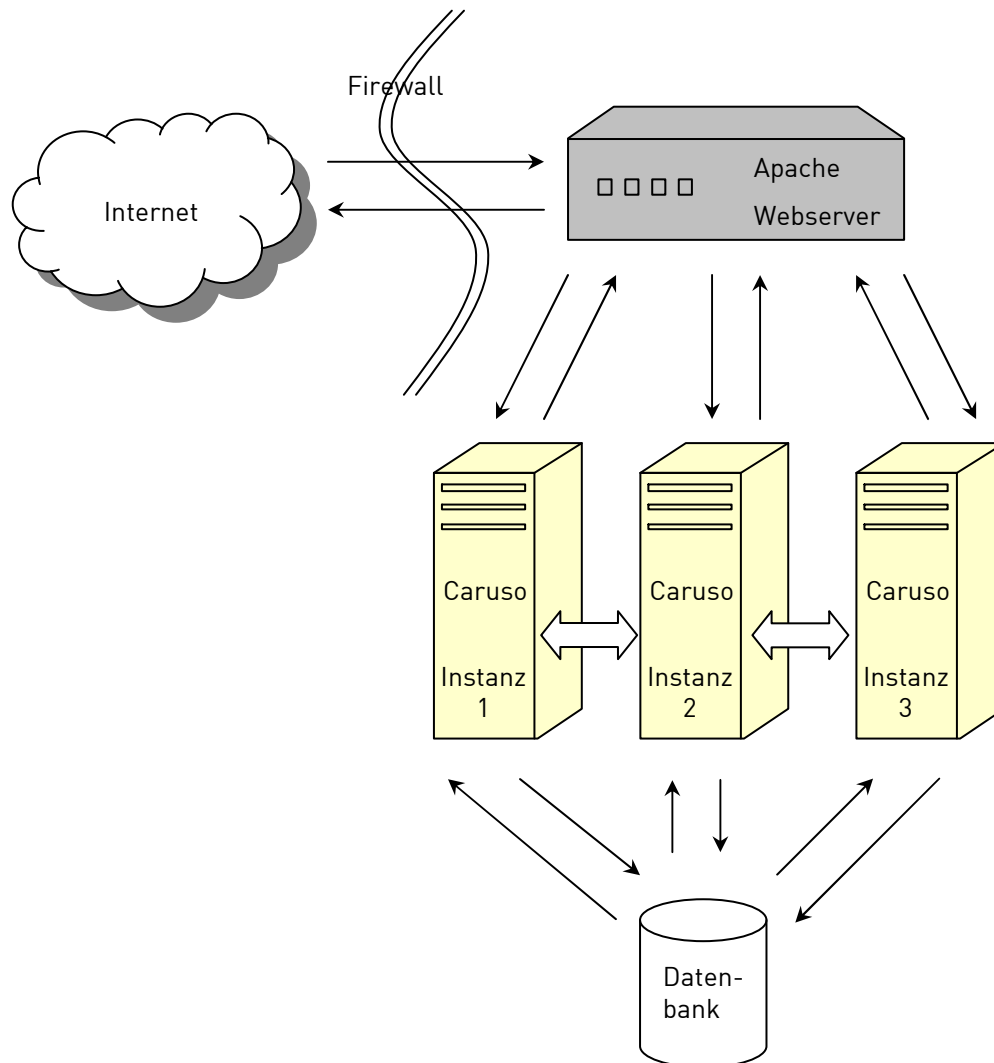
Dieses Beispiel zeigt die Kommunikation von CARUSO mit einer Backupdatenbank, welche nur die für das Internet bestimmten Daten bereithält.



Bei diesem Ansatz steht die Sicherheit im Vordergrund. CARUSO kommuniziert hier mit einer so genannten Backup-Datenbank, welche mittels Replikation ausschließlich mit den für CARUSO notwendigen Daten befüllt wird. Diese Replikation kann direkt über die Datenbanken erfolgen, oder durch eine Replikationsanwendung gesteuert werden. Der Vorteil ist, dass bei einer Sicherheitslücke auf Seiten des Webservers in erster Instanz nur der Backup-Datenbankserver erreichbar wäre. Auf diesem sollten dann auch nur die Daten zur Verfügung stehen, welche im direkten Zusammenhang mit der Darstellung in CARUSO stehen. Ein Nachteil liegt allerdings in dem wesentlich komplexeren Aufbau. Hier werden zusätzlich zu der bestehenden Architektur zwei neue Server benötigt: zum einen der Webserver und zum anderen der Backup-Datenbankserver. Darüber hinaus muss noch eine Kommunikation zwischen dem Produktiv-Datenbankserver und dem Backupserver erstellt werden, was je nach Anforderung beliebig komplex sein kann.

### 3.3 Server-Aufbau für Load-Balancing / Failover

Zusätzlich zu den vorangegangenen Szenarien für die Datenbank-Anbindung ist folgender System-Aufbau für ein Load-Balancing- bzw. Failover-Szenario möglich.



Bei diesem Ansatz wird eine identische Caruso-Installation auf mehreren Rechnern installiert, die alle auf die gleiche Datenbank zugreifen. Für den Failover-Fall werden die aktuellen Session-Informationen der Benutzer automatisch zwischen allen Instanzen synchronisiert, so dass der Benutzer vom Ausfall einer Instanz nichts mitbekommt.

Den Caruso-Instanzen wird ein Apache Webserver vorgeschaltet, der die Lastverteilung übernimmt. Hierbei werden die Web-Anfragen der Benutzer gleichmäßig auf die verfügbaren Instanzen verteilt. Sollte eine Instanz ausfallen, so werden in dem Fall auch keine Anfragen dorthin weitergeleitet.

Um die Ausfallsicherheit weiter zu verbessern, sind zwei weitere Ergänzungen möglich.

Zum einen kann der Apache Webserver ebenfalls redundant betrieben werden, wenn davor ein Hardware- oder Software-LoadBalancer gesetzt wird, der die Anfragen gleichmäßig auf die verfügbaren Apache Webserver verteilt.

Zum zweiten verfügen einige Datenbank-Systeme wie z.B. Oracle (Enterprise) über eigene Möglichkeiten des Clustering bzw. der Lastverteilung und des Failovers, so dass im Normalfall auch hier eine skalierbare Lösung ermöglicht wird.

Neben der dargestellten Aufteilung der einzelnen Instanzen auf unterschiedliche physikalische Rechner ist es, je nach Anforderung, in der Regel auch möglich, mehrere parallele Instanzen auf der selben Hardware zu betreiben. Zwar würde in dem Fall der Ausfall eines Servers auch die Nichtverfügbarkeit von mehr als einer Instanz bedeuten, allerdings nutzt diese Variante die verfügbare Hardware besser aus, da aktuelle Server-Hardware – je nach Anforderung und Ausstattung – durchaus mehr als eine Instanz ausführen kann.