Baden für Anfänger mit ADF Business Components „By the Poolside“

Frank Nimphius
Oracle Corporation
Düsseldorf

Schlüsselworte
Module Pooling, ADF, ADF Business Components, Java EE

Einleitung

Im wirklichen Leben ist ein Pool ein Becken in dem sich gleichgesinnte zum Baden treffen. In einem Leistungszentrum für Schwimmer kann ein Pool aber auch ein Becken sein in dem Schwimmer sich aufhalten bis sie an den Start gehen, was dann den Vorteil hat, dass sie bereits aufgewärmt sind, wenn der Startschuss fällt.

Wie im oben genannten Beispiel ist auch bei dem ADF Business Components (ADF BC) Module Pooling ein Anwendungsmodule (AM) Pool vergleichbar mit einem Aufwärmbecken in dem vor-instanziierte AM bereit gehalten werden. Bei Bedarf erbringen diese Anwendungsmodule dann sehr viel schneller Höchstleitungen, was zu einem besseren Antwortverhalten der Anwendung führt.

Was aber steckt hinter dem AM Pooling und für welche Anwendungen macht das überhaupt Sinn? Wie verhält sich eine Request-übergreifende Transaktion und wie passen Begriffe wie „passivation“ und „activation“ in das Bild?

Das Anwendungsmodul in ADF BC

Anwendungsmodule in ADF Business Components sind Geschäftsmodelle, Blaupausen von denen beliebig viele Instanzen gleicher Konfiguration zur Laufzeit erstellt und verwaltet werden können. Jede Instanz wird dabei einem Anwender zugeordnet.

In diesem Zusammenhang übernimmt das Anwendungsmodul drei wichtige Aufgaben. Zum einen definiert das AM die Programmierschnittstelle für den ADF Entwickler aus den vorhandenen View Objekten, deren relationale Beziehungen zueinander und den öffentlichen Java client Methoden.

Wenn ein Anwendungsmodul selbständig agiert (Root AM), also nicht das untergeordnete Objekt eines anderen Anwendungsmoduls ist, verwaltet es eine eigene Transaktion und bestimmt dessen Grenzen. Alle Datenänderungen in den View Objekt Instanzen einer Root-AM werden gemeinsam festgeschrieben oder zurückgerollt.

Die dritte wichtige Funktion des Anwendungsmoduls ist es sich den Zustand einer Transaktion über einen einzelnen Client-Request hinaus zu merken. Um zum Beispiel einen kompletten Bestellvorgang innerhalb einer Anwendung zu dokumentieren müssen Datenänderungen protokolliert werden bis der Benutzer den Vorgang mit einem Commit oder Rollback abschließt. Ähnlich wie bei einer Datenbank Session dürfen Datenänderungen nur innerhalb der Anwender Session sichtbar sein bis sie committed wurden.

Wenn der Mensch denkt langweilt sich der Server

Um die Skalierbarkeit einer auf ADF Business Components basierten Webanwendung zu bestimmen könnte man aus dem bisher geschriebenen ableiten, dass die Anzahl der möglichen User Sessions, und damit die Gesamtzahl der User auf einem Server, der Division des verfügbaren Arbeitsspeichers durch das Produkt aus den pro Anwender benötigten Root-AMs und deren Speicherbedarf entspricht.

Das wäre richtig gerechnet und dennoch weit gefehlt, da bei dieser Kalkulation das Aufwärmbecken nicht ausreichend berücksichtigt würde. Denken Sie daran, Schwimmwettkämpfe benötigen auch nur acht Bahnen für mitunter 30+ Schwimmer.

Nutzer einer Webanwendung sind keine Maschinen und benötigen Zeit für die Dateneingabe, für das Überlegen, das Suchen und für Pausen. Dadurch entstehen auf dem Server Leerzeiten, die verwendet werden können um Anfragen anderer Anwender abzuarbeiten. Um das zu machen, und dadurch mehr Nutzer auf einem System bedienen zu können als rechnerisch möglich wäre, ist eine Funktionalität erforderlich die Antworten auf folgende Fragen liefert:

1. Wie werden Stoßzeiten in denen überdurchschnittlich viele Aufrufe an den Rechner gesandt werden am besten abgearbeitet?

2. Was passiert mit der Transaktion eines Anwenders zwischen zwei zeitlich versetzen Aufrufen?

3. Wie kann gewährleistet werden, dass geänderte Daten, die noch nicht in der Datenbank gespeichert wurden, nur von dem Anwender gesehen werden können der die Daten geändert?

4. Wie kann ein Datenverlust zwischen zwei Anwendungsaufrufen durch einen Nutzer verhindert werden.

Ich glaube, Sie merken es bereits: Wir nähern uns mit großen Schritten dem Konzept des „Application Module Pooling“ in ADF BC, sowie den Themen „Passivation“ und „Activation“.

Ein Pool, viele Schwimmer

In öffentlichen Freibädern gibt es eine Obergrenze für die Anzahl von Gästen die sich zeitgleich im Bad aufhalten dürfen. Diese Obergrenze liegt in der Regel deutlich über der Anzahl von Schwimmern die zeitgleich in den Pool passen, ohne dass aus Schwimmen nur noch Plantschen wird. In öffentlichen Freibädern geht man also davon aus, dass ungeachtet der Hitze nicht alle Schwimmer zur gleichen Zeit das Schwimmbecken aufsuchen. Das gleiche Prinzip findet auch bei ADF Business Components statt:

Abhängig von der Konfiguration eines Anwendungsmodul werden von einem ADF BC Root-AM zur Laufzeit unterschiedlich viele Instanzen gestartet und im Arbeitsspeicher des Servers vorgehalten. Der reservierte Speicherplatz wird von ADF Business Components für das jeweilige Anwendungsmodul verwaltet und als AM Pool bezeichnet.

Um den Arbeitsspeicher nicht unnötig zu belasten kann der AM Pool so konfigurieren werden, dass ab einer bestimmten Schwelle, die fernab der maximalen Kapazität von Anwendungsmodul-Instanzen liegt, jene Instanzen die schon länger nicht verwendet wurden einem anderen Anwender zugeordnet werden. Das ist „preiswerter“ als neue Instanzen so lange zu starten bis die maximale Kapazität für einen Server erreicht ist.

An dieser Stelle ist der Unterschied zwischen dem Freibad und ADF BC, dass Badegäste als „ user“, Schwimmer als „concurrent user“ und die einzelne Schwimmeinheit als Session bezeichnet werden.

Die „Schwelle“ an der in ADF BC Anwendungsmodul Instanzen an andere Anwendersession gegeben werden ist konfigurierbar und mit „Recycle Treshold“ bezeichnet.

Im Vergleich zu dem öffentlichen Bäderbetrieb entspräche die „Recycle Treshold“ dem Fall in dem der Bademeister ab einer bestimmten Anzahl von Personen im Becken die am Rand „nur“ badenden Besucher aus dem Becken bittet um Platz für echte Schwimmer zu schaffen.

Um eine ADF BC Modul-Instanz für eine andere Anwendersession zu verwenden muß der Zustand der jeweiligen AM Instanz verlässlich gespeichert werden für den Fall daß der letzte Anwender dieser Instanz in einem Folgerequest seine Arbeit fortsetzen möchte.

Der Prozeß der den Zustand einer AM Instanz für später speichert wird als „Passivation“ bezeichnet.

Sie können sich vorstellen, daß der Prozeß der „Passivation“ I/O intensiv ist und je nach Speicherort auch das Netzwerk belastet. Selbiges gilt für die Wiederherstellung einer Instanz aus dem Speicherort, der sogenannten „Activation“.

Zu diesem Zeitpunkt können wir bereits den Schluß ziehen, daß die optimale Performance einer ADF Business Components Anwendung erreicht wird indem Sie das AM Pooling so konfigurieren, dass der Arbeitsspeicher nicht zu sehr belastet und unnötige „Passivation“ und „Activation“ Zyklen vermieden werden.

Hinweis**:** In Produktivumgebungen ist ein Betrieb von ADF BC ohne AM Pooling nicht supported.

Das „Verstehen“ als Schwimmabzeichen

Um zu wissen, wie viele Anwendungsinstanzen benötigt werden um „Passivation“ und „Activation“ Zyklen gering zu halten, ist es wichtig zu verstehen wie Module Pooling funktioniert und wann und wie eine Anwendungsmodulinstanz einer Anwendersession zugeordnet wird.

Ein ADF Business Components Anwendungsmodul kann drei verschiedene Zustände zur Laufzeit annehmen: verfügbar, nicht verfügbar, referenziert.

Wenn der AM Pool für ein Anwendungsmodul gestartet wird dann werden automatisch eine Menge „x“ an AM Instanzen mit gestartet. Die Anzahl „x“ wird durch den „initial poolsize“ Parameter in bestimmt und sollte ausreichend groß gewählt werden um den zu erwartenden ersten „Ansturm“ von Anwender-Sessions zu bewältigen.

Wenn ein Anwendungsmodul aus dem Pool genommen wird, so ändert sich der Verfügbarkeitsstatus auf „nicht verfügbar“. Das Module wird also nicht aus dem Pool entfernt sondern nimmt einen neuen Zustand ein: „bin beschäftigt“.

Wird eine zuvor „beschäftigte“ AM Instanz frei, so ändert sich der Zustand auf „referenziert“, was Sie sich als „reserviert“ vorstellen können. Das Anwendungsmodul behält dabei seinen derzeitigen Query und Transaktionszustand für den Fall, dass der selbe Anwender eine Folgeanfrage stellt. Der Zustand des Moduls wird im Pool „stateful“ gehalten.

Im besten Fall kann somit ein Folgeaufruf von der selben AM Instanz abgearbeitet werden, was zu einem optimalen Antwortverhalten führt.

Konkret erfolgt de Vergabe von Anwendungsmodul Instanzen nach den folgenden Regeln: Zunächst einmal prüft der Pool Manager ob ein referenziertes Module für einen eingehenden Request verfügbar ist, was immer die beste Option ist.

Handelt es sich um einen Request der keiner referenzierten oder passivierten AM Instanz zugeordnet werden kann, so wird zunächst geprüft, ob eine „verfügbare" AM Instanz im Pool vorhanden ist. Nur wenn das nicht der Fall ist, wird die „Recycle Treshold" Schwelle geprüft um entweder eine neue AM Instanz zu starten oder eine bestehendes Instanz, nach dessen Passivierung, wiederzuverwenden.

Bei der Passivierung wird immer die Anwendungsinstanz gewählt die sich am längsten tätigkeitslos im Pool befindet. Bei den öffentlichen Bäderbetriebe wären es die Badenden, die sich an den Ränder des Beckens klammern um sich, zum Beispiel über die herbstliche Laubverbrennung zu unterhalten.

Mit dem bisher geschriebenen kann über das ADF BC Module Pooling somit gesagt werden, dass das ADF BC Framework zu jeder Zeit sicher stellt, dass eingehende Anfragen für ein AM Modul mit der jeweils besten möglichen Performance beantwortet werden.

Die Einstellmöglichkeiten zum Erzielen bestmöglicher Performance beinhaltet die Größe des Pools (Max-Size), die Mindestanzahl von AM Instanzen die zu jedem Zeitpunkt im Pool sein müssen (Min-Size) und die Bestimmung der „Recycle Treshold" Schwelle oberhalb derer AM Instanzen passiviert werden.

Was die Max-Size des Pools angeht, so können Sie ungefähr davon ausgehen, daß eine AM Instanz zwischen 10-15 Anwendersessions sequentiell abarbeiten kann. Dieser Empfehlung liegt dabei der Grundsatz „Wenn der Mensch denkt langweilt sich der Server„ zugrunde.

Was kostet „Passivation" und „Activation"

„Passivation" und „Activation" sind wichtige Eigenschaften des AM Pooling und ermöglichen den Zustand von Anwendungsmodul-Instanzen festzuschreiben damit diese Anfragen anderer Anwender Sessions bearbeiten können. Was während der „Passivation" gespeichert wird und wo es gespeichert wird soll im folgenden helfen die Frage der mit der „Passivation" verbundenen „Kosten" zu klären.

Bei der „Passivation" eines Anwendungsmodules werden Informationen die zur Wiederherstellung der Anwendersession benötigt werden in XML innerhalb der Datenbank der dem Dateisystem gespeichert. Standardmäßig wird die Datenbank zur Speicherung der passivierten Informationen verwendet wenn die Datenbank eine Oracle Datenbank oder DB2 ist. Bei anderen Datenbanken wird das Dateisystem als voreingestellter Speicherort verwendet. Das alles ist natürlich vom Entwickler frei konfigurierbar.

Im wesentlichen fallen die „Kosten" der „Passivation" durch das Schreiben des Session-Zustandes nach XML und dem späteren lesen, bei der „Activation", aus dem XML Format an.

Das Schreiben in die Datenbank fügt dem, im Vergleich zum einfachen I/O des Dateisystems, einen kleinen Aufpreis hinzu.

Zu den Informationen die zur Wiederherstellung der Anwendersession gespeichert werden gehören der aktuelle Zeilencursor in einem Row-Set, neue, geänderte oder gelöschte Datenobjekte, Query-

Filter, Parameter, Transiente View Objekte soweit konfiguriert, sowie weitere Informationen zu den View Objekt Konfigurationen .

Das was bei der „Passivation“ nicht gespeichert wird sind Datensätze, die über die View Objekte aus der Datenbank gelesen, seit dem aber nicht geändert oder gelöscht wurden. Da diese Information bei der „Activation“ durch eine neue Datenbankabfrage neu gelesen wird kann es dazu kommen, daß ein Anwender in der Datenbank geänderte und neue Datensätze sieht, die er oder sie vor der „Passivation“ nicht gesehen hat. ADF Business Components ist bei der „Activation“ also nicht Lesekonsistent. Um Irritierungen der Anwender zu vermeiden sollten daher alle Entity- und View Objekte zumindest ein Attribut als Primärschlüssel gekennzeichnet haben.

Wenn Sie also nach Gründen suchen “Passivation” und “Activation” zu vermeiden, dann haben sie gleich drei gneannt bekommen

1. Kosten der Serialisierung der zu speichernden Informationen nach XML
2. Kosten durch I/O Operationen im Dateisystem oder der Datenbank
3. Leseinkonsistenz beim Wiederherstellen einer Anwender Session

Die Kunst beim AM Pool Tuning besteht im Grunde darin den ADF Business Components AM Pool so zu dimensionieren, daß “Passivation” nicht die Regel, sondern die Ausnahme ist. Im übetragenen Sinne, und um mal wieder den Vergleich zu dem öffentlichen Bad zu bemühen, soll “Passivation” nur den “Badenden”, nicht aber den “Schwimmer” treffen.

Hinweis**:** Ein Usecase jenseits des AM Pooling in dem “Passivation” verwendet wird ist “Fail over”. Die Informationen, die bei der “Passivation” gespeichert werden erlauben es den AM Zustand einer Anwender Session auch auf einem anderen Knoten innerhalb eines Clusters wieder herzustellen. “Fail over” ist ein sogenanntes “opt-in”, muß also durch Konfiguration extra aktiviert werden.

Zusammenfassung

AM Pooling in ADF Business Components verbessert die Skalierbarkeit von Java EE Anwendungen. Dabei ist es wichtig den Pool so zu dimensionieren, daß zu jeder Zeit genügen – aber auch nicht zuviel – AM Modul Instanzen verfügbar sind um aufeinanderfolgende Anfrage einer Anwendersession von derselben AM Modul Instanz bearbeiten lassen zu können.

Für Anwendungen die den Zustand einer Transaktion über mehrere Anwender-Requests hinaus halten müssen wird neben einem ausreichend dimensionierter AM Pool noch “Passivation” und “Activation” verwendet. Beides, AM Pooling und “Passivation” und “Activation” sind standardmäßig für ADF BC Anwendungen aktiviert.
Die richtige Dimensionierung eines AM Pools ist abhängig von der Verwendung des Anwendungs-moduls, also dessen Usecase, sowie der zu erwartenden Zahl der Anwenderzugriffe, und kann daher nicht generalisiert für alle Anwendungen bestimmt werden.

Kontaktadresse:
Frank Nimphius
Oracle Corporation
Hamborner Straße 51
40472 Düsseldorf

E-Mail frank.nimphius@oracle.com