Archivsystem Release-Notes

Jonathan Haas, Roland Schmitz

5. Oktober 2017

Kappich Systemberatung

Inhaltsverzeichnis

1 Einleitung

Im folgenden Dokument werden die Release-Notes zum Archivsystem in umgekehrter chronologischer Reihenfolge aufgelistet, so dass die Informationen zum letzten Release zuerst aufgeführt werden.

2 Archivsystem 3.10.0 Beta 2

Release-Datum: 5.10.2017

2.1 Änderungen

- Die neue Version des Archivsystems benötigt als zusätzliche Abhängigkeit die Kernsoftware in Mindestversion 3.9.6. Die Archiv-Fehler Nerz-F–44 und Nerz-F–141 wurden in dieser Kernsoftwareversion korrigiert.
- Nerz-Ä–8: Die Anzahl der gleichzeitig geöffneten Container-Dateien beim Lesen von Archivdaten wird auf standardmäßig 500 begrenzt um Limits des Betriebssystems zu umgehen. Hierzu wurde eine Zugriffsschicht implementiert, die beim Überschreiten des Grenzwertes die älteste geöffnete Containerdatei schließt. Bei Bedarf lässt sich mit den neuen Aufrufargument `-maxOffeneDateien=500` ein anderer Grenzwert vorgeben.
- Nerz-Ä–10: Das Archivsystem veröffentlicht jetzt den verfügbaren Speicherplatz im Persistenzverzeichnis am Archiv-Objekt unter der Attributgruppe `atg.archivSpeicherplatz`.
- Nerz-F–49: Beendet sich das Archivsystem aufgrund eines Fehlers, wird nun über die Debug-Ausgabe ein Thread-Dump ausgegeben, um Fehler besser diagnostizieren zu können.
- Nerz-F–55: Die Implementierung des Backup-Indexes wurde angepasst, sodass das Einfügen von doppelten Container-IDs im Code mit einer Fehlermeldung verhindert wird.
- Nerz-F–91: Folgende zusätzliche Hintergrund-Tasks werden beim Beenden des Archivsystems nun zeitnah beendet (ohne Datenverlust):
  - Sichern
  - Archivanfragen
  - Archiv-Informations-Anfragen

Beim Sichern wird das aktuelle Medium mit den ggf. bereits gesicherten Daten abgeschlossen, weitere zu sichernde Daten werden dann erst beim nächsten Sicherungslauf gesichert. Die Archivanfragen werden abgebrochen, sodass der Anwender beim Abruf eine entsprechende Fehlermeldung erhält.

- Nerz-Ä–66: Die Anzahl der Threads zur Archivierung für Online-Datensätze kann nun mit dem Aufrufargument `-archivierungsThreads=` angegeben werden. Der Standardwert wurde von 61 auf 11 geändert. Die Anzahl Threads zur Archivierung von nachgeforderten Datensätzen lässt sich mit `-nachforderungsThreads=` beeinflussen. Der Standardwert hierfür ist jetzt 3 statt vorher 8.

  Zusätzlich kann nun die von allen Threads geteilte Puffergröße durch ein Aufrufargument angepasst werden. Frühere Archivsystem-Versionen besaßen eine feste Gesamtgröße von 183.000 für Onlinedaten und eine Gesamtgröße von 16.000 für nachgeforderte Daten. Die Puffergröße kann nun durch die folgenden Aufrufargumente angepasst werden (mit neuen Standardwerten):

  ```
  -archivierungsPuffer=200000
  -nachforderungsPuffer=20000
  ```

  Mit einer größeren Puffergröße kann (zusammen mit den Änderungen aus Nerz-Ä–75) das Terminieren der Verbindung wegen zu geringem Durchsatz während Lastspitzen vermieden werden.

- Nerz-Ä–68: Das Schreiben der Indexdateien beim Beenden erfolgt nun in mehreren Threads gleichzeitig. Die Threadanzahl entspricht standardmäßig der Anzahl Threads für die Online-Archivierung (Aufrufargument `-archivierungsThreads=11` aus Nerz-Ä–66), kann jedoch mit dem Aufrufargument `-indexUpdateThreads=` überschrieben werden. Es erfolgt nun alle 10 Sekunden eine Debug-Ausgabe über den Fortschritt.

- Nerz-Ä–69: Die manuelle Konsistenzprüfung erfolgt nun in mehreren Threads gleichzeitig. Die Threadanzahl beträgt standardmäßig der Anzahl Threads für die Online-Archivierung (Aufrufargument `-archivierungsThreads=11` aus Nerz-Ä–66), kann jedoch mit dem Aufrufargument `-pruefThreads=` überschrieben werden. Es erfolgt nun alle 10 Sekunden eine Debug-Ausgabe über den Fortschritt.

- Nerz-Ä–70: Das Sichern von Containern wurde durch Verwendung eines `BufferedInputStream`s und einer größeren Puffergröße von 8 kB optimiert.

- Nerz-Ä–75: Die Anmeldungen von Daten nach dem Start des Archivsystems oder bei einer Parameteränderung werden nun automatisch gedrosselt, um eine Überlastung des Systems zu vermeiden. Hierzu wurde ein Sliding-Window-Mechanismus implementiert, der die Anzahl noch nicht quittierter Anmeldungen beschränkt und weitere Anmeldungen erst dann zulässt, wenn schon durchgeführte Anmeldungen (durch den initialen Datensatz) quittiert werden.

Bei Bedarf lässt sich der Mechanismus über folgende neue Aufrufparameter (Standardwerte sind angegeben) einstellen:

```
-maxAnmeldungenGleichzeitig=1000
-minAnmeldungenProSekunde=1
-maxAnmeldungenProSekunde=<Unendlich>
```

`maxAnmeldungenGleichzeitig` definiert die Größe des Sliding-Windows, also die maximale Anzahl gleichzeitig unbestätigter Anmeldungen. Mit zum Beispiel `-maxAnmeldungenGleichzeitig=1` würde das Archivsystem immer nur eine Anmeldung gleichzeitig durchführen und dann warten bis diese bestätigt wurde, bevor die nächste Anmeldung durchgeführt wird.

Mit `minAnmeldungenProSekunde` lässt sich der Sliding-Window-Mechanismus beschleunigen, beispielsweise würde `-minAnmeldungenProSekunde=2000` dafür sorgen, dass pro Sekunde maximal 2000 Anmeldungen durchgeführt werden, selbst wenn es mehr als `maxAnmeldungenGleichzeitig` unbestätigte Anmeldungen gibt und die Anmeldung dadurch verzögert worden wäre.

Mit `maxAnmeldungenProSekunde` lassen sich die Anmeldungen unabhängig vom Sliding-Window-Mechanismus verzögern. Beispielsweise würden mit `-maxAnmeldungenProSekunde=10` maximal 10 Anmeldungen in der Sekunde durchgeführt werden, unabhängig von allen anderen Faktoren. Diese Funktion ist dadurch implementiert, dass das Archivsystem zwischen den Anmeldungen eine definierte Zeit wartet. Die hierzu verwendete Funktion der virtuellen Maschine besitzt nur eine eingeschränkte Genauigkeit, die auch stark vom verwendeten Betriebssystem abhängt, wodurch unter Umständen eine weitere unbeabsichtigte Verzögerung der Anmeldungen entstehen kann, das Setzen von `-maxAnmeldungenProSekunde=100000` oder einem ähnlich hohen Wert führt daher wahrscheinlich zu einer unerwartet hohen Verzögerung und ist nicht empfehlenswert. Falls es Überlastungsprobleme gibt, ist in der Regel das Reduzieren von `maxAnmeldungenGleichzeitig` ausreichend, die anderen Parameter sollten daher nur in Ausnahmefällen benutzt werden.

- Nerz-Ä–79 (teilweise): Das ContainerRescue-Programm erkennt jetzt unvollständig geschriebene (in der Regel noch nicht abgeschlossene) Containerdateien und kann überflüssige Bytes am Ende sowie unvollständige Datensätze erkennen und entfernen.

2.2 Bugfixes

Folgende Probleme vorhergehender Versionen wurden behoben:

- Nerz-F–2: Falls ein Container nicht gesichert werden konnte, weil die Datei nicht vorhanden ist, wird nun eine klare Fehlermeldung erzeugt und der Container zukünftig ignoriert (aus dem Index entfernt). In vorherigen Versionen kam es bei jedem Backup zu Fehlermeldungen, da der gelöschte Container im Index blieb.
- Nerz-F–4: Ein Problem, bei dem während des spontanen Löschens im Lösch-Level 2 alle abgeschlossenen Container gelöscht wurden, wurde korrigiert. In vorigen Versionen wurden wegen fehlender `break;`-Statements in einem `switch`-Block schon in Lösch-Level 2 die Lösch-Verfahren aus Level 3 und 4 angewendet. Außerdem wurden im Lösch-Level 4 wegen eines fehlerhaften String-Vergleichs alle abgeschlossenen Container gelöscht, nicht wie spezifiziert alle bereits gesicherten.
- Nerz-F–36: Die Queue-Füllstände für nachgeforderte Datensätze werden in den periodischen Debug-Ausgaben nun getrennt von den Online-Daten aufgeführt und auch im Code besser voneinander unterschieden.

  Das Nachfordern von Daten führte wegen Fehlern bei der Verwendung von globalen Statistik-Variablen zu negativen Angaben der Queue-Größe in den Debug-Meldungen.
- Nerz-F–37 /Nerz-F–164: Das Archivsystem erstellt keine überflüssigen temporären Container-Dateien mehr, wenn Daten wiederhergestellt werden sollen. In alten Versionen konnte es zu Problemen kommen, wenn der Speicherort nicht beschreibbar war oder wenig freier Festplattenplatz zur Verfügung stand.
- Nerz-F–38: Das manuelle Löschen über die DAF-API-Funktionen war fehlerhaft implementiert. Das sofortige Löschen (deleteImmediately==true) bewirkte intern ein Anstoß der spontanen Löschfunktion, die bei ausreichend verfügbarem Speicherplatz keine Daten löscht. Das verzögerte Löschen (deleteImmediately==false) bewirkte stattdessen einen sofortigen Anstoß der regulären periodischen Löschfunktion, die in der Regel ebenfalls nichts gelöscht hat, da sie den Löschschutz betrachtet.
- Nerz-F–90 /Nerz-F–193 /Nerz-F–205: Es ist nicht mehr möglich, das Archivsystem mehrfach zu beenden. Hierzu wurde die Synchronisierung beim Beenden überarbeitet.
- Nerz-F–101 /Nerz-F–188 /Nerz-F–189 /Nerz-F–203: Tasks, die vorzeitig terminiert wurden, führen nicht mehr zu einem Blockieren beim Beenden. Der Code wurde korrigiert, indem die nicht korrekt implementierte Semaphore durch eine einfache Abfrage der Thread-Zustände ersetzt wurde.

  Zusätzlich wurde der Mechanismus zum Starten und Beenden der Tasks komplett überarbeitet. Es kann nun nicht mehr passieren, dass sich ein Task wegen einer unbehandelten Exception oder wegen eines Interrupts ohne Rückmeldung terminiert. Ein Fehler in einem Task führt bei einem schweren Fehler nun zum kontrollierten

Beenden des Archivsystems. Andere Fehler führen zu einer Debug-Ausgabe und lassen den Task weiterlaufen.

Es wurde ein UncaughtExceptionHandler implementiert, der auch außerhalb von Tasks dafür sorgt, dass unbehandelte schwerwiegende Fehler (z. B. `OutOfMemoryError`s) zu einem sofortigen Beenden des Archivsystems führen. Andere unbehandelte Fehler führen zu einer Debug-Warnung.

- Nerz-F–150: Die Ergebnisse von Archiv-Informations-Anfragen wurden grundsätzlich überarbeitet, insbesondere wurden folgende Fehler korrigiert:
  - Es wurden immer Ergebnisse für alle Datenarten zurückgeliefert, unabhängig von den angefragten Datenarten
  - Es konnte zu Performanceproblemen kommen, wenn nicht alle Datenarten angefragt wurden.
  - Das Behandeln von Datenlücken und das Mischen der einzelnen Datenidentifikationen wurde überarbeitet und verhält sich jetzt analog zu den Archivanfragen.
  - Nachgelieferte Daten werden nun (bei entsprechender Anfrage) korrekt einsortiert und führen nicht mehr zu erkannten Datenlücken.
  - Das Archivierungs-Bit in den Datenindizes wurde bei Datenlücken nicht immer korrekt gesetzt.
- Nerz-F–156: Bei der Wiederherstellung von Containern wurde der Sekunden-Wert der Löschschutzverlängerung fälschlicherweise als Millisekunden interpretiert, wodurch der Löschschutz-Zeitraum von wiederhergestellten Containern um den Faktor 1000 zu klein war. Hierdurch wurden beim regulären Löschen häufig wiederhergestellte Container gelöscht, die laut Parametrierung eigentlich einen Löschschutz haben müssten.

  Bei Archivanfragen und Archiv-Info-Anfragen war außerdem der Standardwert der maximalen Löschschutzverlängerung in der Klasse `QueryTask` in Millisekunden definiert, er wird aber aber als Sekundenwert interpretiert. Der um den Faktor 1000 zu hohe Wert führte dabei zu einem Überlauf bei der Verarbeitung als 32-bit-Wert.
- Nerz-F–197: Das Archivsystem kann sich jetzt auch dann aus einer Überlastsituation befreien, bei der Archivanfragen angehalten wurden, wenn keine neuen Datensätze eintreffen. In älteren Versionen wurde die Prüfung, ob die Überlastsituation vorbei ist und die Bearbeitung der Anfragen fortgesetzt werden soll, nur beim Eintreffen eines neuen, nicht ignorierten, Archiv-Datensatzes vorgenommen.

- Nerz-F–209: Die Applikations-Fertigmeldung des Archivsystems wird jetzt erst versendet, wenn zum ersten mal alle zu archivierenden Daten angemeldet wurden. Vorher wurde sie direkt beim Programmstart versendet. Archivanfragen können erst dann durchgeführt werden, wenn alle Daten angemeldet sind.
- Der Wiederherstellungsauftrag hat die vom Anwender übergebenen Minimum- und Maximum-Werte zur Angabe des Zeitbereichs immer als Archivzeitstempel interpretiert, auch wenn vom Anwender Datenindex- oder Datenzeitwerte übergeben wurden. Unter Umständen wurden dadurch nicht die gewünschten Container wiederhergestellt.
- Das Archivsystem hat Archivanfragen sofort verworfen, wenn eine negative Sendesteuerung vor dem Senden der Antwort registriert wurde. Das konnte dazu führen, dass Archivanfragen nicht beantwortet wurden und Aufrufe der Methode ArchiveQueryResult.isRequestSuccessful() blockierten. Da die Sendesteuerung insbesondere bei der Kopplung von Datenverteilern verzögert eintreffen kann, wird nun auf die Sendesteuerung mit einem Timeout gewartet. Sollte eine bereits positiv quittierte Übertragung unterbrochen werden (durch eine negative Quittung), wird wie bisher die Übertragung der Antwort sofort abgebrochen.
- Beim Wiederherstellungslauf nach einem Archivsystemabsturz konnte es aufgrund eines ungünstig programmierten rekursiven Algorithmus zu einer StackOverflowException kommen, wenn im Persistenzverzeichnis viele Ordner ohne Archivdatenart-Unterordner (“oa”, “on”, “na”, “nn”) existieren. Eine solche Ordnerstruktur kann beispielsweise entstehen, wenn Daten nachgefordert werden sollen, die in beiden beteiligten Systemen noch nie archiviert wurden. Der Algorithmus in der Klasse ContainerFileDirIterator wurde in einen iterativen Algorithmus umgewandelt, wodurch der Fehler trotz ungünstiger Ordnerstruktur nicht mehr auftreten kann. Die betroffene Klasse wurde ebenfalls beim Backup, Löschen, Löschzeitverlängern, Archivdaten aus einer Sicherung Wiederherstellen und bei der Konsistenzprüfung verwendet, sodass es dort möglicherweise zu ähnlichen Exceptions kommen konnte.

3 Archivsystem 3.9.0

Release-Datum: 25.02.2014

3.1 Neue Features

Folgende Erweiterungen gegenüber vorhergehenden Versionen wurden durchgeführt:

- Nerz-Ä–12: Begrenzung der Anzahl parallel zu verarbeitenden Archivanfragen seitens einer einzelnen Applikation.
  - Im Archivsystem kann die maximale Anzahl von gleichzeitig für eine Applikation bearbeiteten Archivanfragen über den Aufrufparameter `-maxAnfragenProApplikation` eingestellt werden. Defaultwert ist 5.
  - Über eine entsprechende Archivanfrage kann der aktuell eingestellte Wert zur Laufzeit seitens einer Applikation abfragt werden.

3.2 Änderungen

Folgende Änderungen zu vorhergehenden Versionen wurden durchgeführt:

- Nerz-Ä–16: Performance-Verbesserungen bei der Verwendung von ByteArrays in den Datenverteiler-Applikationsfunktionen. Verwendung der neuen Methoden des Data-Interface zum Setzen von `byte[]`, `short[]`, `int[]` etc.

3.3 Bugfixes

Folgende Probleme vorhergehender Versionen wurden behoben:

- Bei Archivanfragen mit relative Zeitangaben konnte es unter bestimmten Umständen passieren, dass das Archivsystem in einer Endlosschleife hängen blieb.