Gut Wetter machen!

Java, Play und Akka für meteorologische Anwendungen beim Deutschen Wetterdienst

Martin Lehmann, Marcus Werner

Java, Akka und Play kommen beim Deutschen Wetterdienst für das operationelle meteorologische Arbeitsplatzsystem und für einen Archivzugriff zum Einsatz. Die Herausforderungen dabei: sehr große Datenmengen, sehr heterogene Datenformate, wenig verbindliche Standards, hohe fachliche Komplexität von Daten und Metadaten, sehr heterogene zeitliche Auflösungen, mangelnde Datenqualität von „Altdaten“.

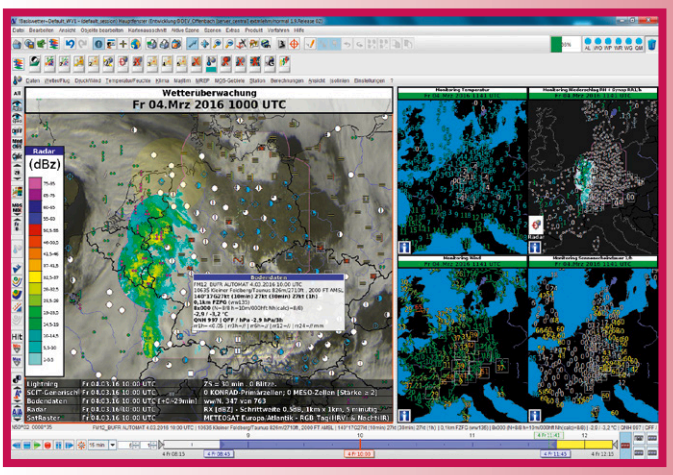

Abb. 1: Der NinJo-Client zeigt einen kühlen und verregneten Märztag: Die Deutschlandkarte links stellt u. a. dar: Satellitenbild im Hintergrund, darüber Radardaten, darüber Symbole für Messdaten

NinJo macht Wetter begreifbar!

Die moderne Wettervorhersage basiert darauf, dass der Meteorologe unterschiedlichste Daten effizient in seinen Vorhersageprozess einbindet. Wegen Datenvielfalt und enormer Datenmengen (ca. 1 TB/Tag) kann er diese Informationen heutzutage nur mit optimierten grafischen Systemen nutzen.

Die meisten von uns verbinden mit dem Wetter die tägliche Wettervorhersage, die über Fernsehen, Radio, Presse, Internet und App veröffentlicht wird. Darüber hinaus übernimmt der Deutsche Wetterdienst (DWD) im Rahmen gesetzlich geregelter Aufgaben die Informierung und individuelle Beratung von Organisationen, Behörden und Firmen: Schlüsselkunden kommen aus Luftrettung, Katastrophenschutz, Verkehr (Straße, Luftfahrt, Schiene, Schifffahrt), Bau- und Energiewirtschaft, Land- und Wasserwirtschaft.

Im DWD kommt für diese Aufgaben seit Jahren das meteorologische Arbeitsplatzsystem NinJo zum Einsatz [HeHa09], das deutschlandweit auf ca. 450 Arbeitsplätzen operativ ist. Zur Reduktion von Entwicklungs- und Wartungskosten erstellt der DWD dieses System in einem Konsortium gemeinsam mit den Wetterdiensten der Schweiz, Dänemarks, Kanadas und der Bundeswehr. Diese Zusammenarbeit soll bewusst den fachlichen Austausch zwischen den Wetterdiensten fördern. Das gesamte internationale NinJo-Projektteam umfasst ca. 80 Personen. Des Weiteren wird mit verschiedenen Softwarehäusern zusammengearbeitet, um flexibel und schnell auf eine breite Palette technischen Know-hows zurückgreifen zu können: Accso unterstützt die Entwicklung am Standort Offenbach. NinJo ist heute bei acht Organisationen auf drei Kontinenten auf mehreren Tausend Workstations operativ und hat prominente Hochlast-Einsätze wie den bei den Olympischen Winterspielen 2010 in Vancouver erfolgreich absolviert.

NinJo ist als Client-Server-System konzipiert und komplett in Java entwickelt:

- Der NinJo-Client kümmert sich um Nutzer-Interaktion und um die Visualisierung der verschiedenen Daten (typischer Screenshot s. Abb. 1). Die grafische Benutzeroberfläche ist mit Swing umgesetzt, Visualisierungen erfolgen unter anderem mit Java 2D, Java 3D sowie Java Advanced Imaging. Der Meteorologe kann die verfügbaren Daten frei kombinieren, zu- oder abschalten und ihre grafischen Attribute vielseitig konfigurieren (Farbtabellen, Größe, Abstände, Datenbereiche, Einheiten usw.).
- Spezielle NinJo-Datenserver halten je nach Datenart, -frequenz, -struktur und Zugriffsvarianten Daten für die Clients bereit. Im DWD kommen ca. 130 Datenserver an ca. 20 Standorten sowohl in der Zentrale Offenbach als auch in sechs Regionalzentralen zum Einsatz. Aktuelle meteorologische Daten werden im deutschlandweiten DWD-Netz redundant verteilt. Durch diese räumliche Aufteilung von Servern und Daten können trotz hoher Datenmengen gute Antwortzeiten und eine sehr hohe Ausfallsicherheit erreicht werden.

Das NinJo-Batchsystem erzeugt außerdem täglich ca. 120.000 Visualisierungen als Bilddateien (PNG, TIFF usw.).

Herausforderung: sehr heterogene Daten

Meteorologische Daten sind auf verschiedene Arten heterogen:

- viele unterschiedliche Datenquellen,
- heterogene fachliche Strukturen (Beispiel: flächige Rasterbilder von Satelliten sind anders zu behandeln als punktbezogene Messdaten von Messstationen),
- verschiedene technische Dateiformate in der Anlieferung,
- heterogene zeitliche Frequenzen der Daten: Manche Daten treffen regelmäßig und kurzgetaktet ein (Beispiel: Radarmessung im 5-Minutentakt), andere haben größere Taktung bei viel größeren Datenmengen (Beispiel: numerische Modellvorhersage), wieder andere Daten treffen nur selten und ereignisbasiert ein (Beispiel: Blitzmessung).
- Die Datenvielfalt zeigt nicht notwendigerweise ein konsistentes Gesamtbild. Deswegen erstellt eine Gruppe erfahrener Meteorologen eine Gesamtbewertung und eine aktuelle Vorhersage für verschiedene zeitliche und räumliche Gebiete.

Dies alles bedingt hohe nicht-funktionale Anforderungen, vor allem hinsichtlich Durchsatz und Performanz: Messungen müssen schnellstmöglich sichtbar sein – so dürfen beispielsweise nur Sekunden vergehen zwischen Blitzeinschlag und Darstellung auf dem Nutzerbildschirm!

Big Data mal anders: Daten aus vielen Quellen

Kernaufgaben des DWD sind Wetterüberwachung, Ausgabe von Wetterwarnungen und Erstellung von Wettervorhersagen. Dazu nutzen Meteorologen aktuelle Messungen (ca. 36 Stunden zurück) und aktuelle Vorhersagedaten (ca. 10 Tage in die Zukunft).

Tabelle 1 zeigt nur einen kleinen Ausschnitt: Daten kommen nicht nur aus dem Bodenmessnetz, sondern auch von Radioaktivitätsmessstationen, von phänologischen Beobachtungsstellen, von Hunderten Wettermeldestellen auf Handelsschiffen usw. Außer von Satellit und Radar kommen Fernerkundungsdaten u.a. auch von Blitzsensoren, Lasermessverfahren, Tausenden Ballonaufstiegen beziehungsweise Flugzeugmessungen. Abgeleitete Produkte entstehen nicht nur im Modell-Supercomputer, sondern auch in nachgelagerten meteorologischen Verfahren für gezielte Analysen, die Daten einsammeln, aggregieren und als neue Produkte ausgeben (Beispiel: Warninformationen für Gefahrenlagen wie Gewitter, Nebel, Glatteis).

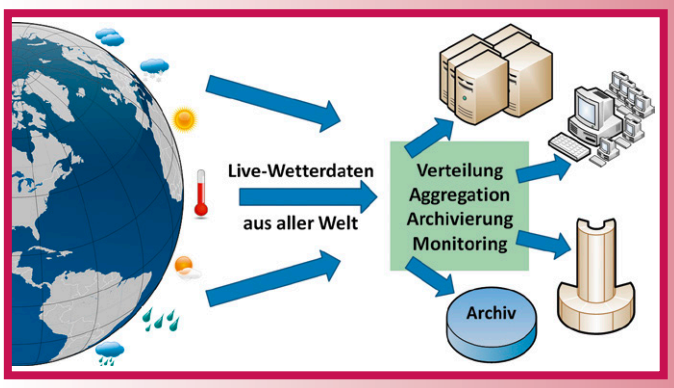

Abb. 2: Im Live-Betrieb treffen kontinuierlich Wetterdaten aus ca. 190 Ländern der WMO ein, darunter von etlichen Tausend Wetterstationen

Weltweiter Datenaustausch zwischen Wetterdiensten

Das Wetter macht nicht an staatlichen Grenzen halt. Daher kooperieren die Wetterdienste einzelner Länder eng und tauschen weltweit Mess- und Vorhersagedaten aus. Sie sind über die World Meteorological Organization (WMO, Sonderorganisation der Vereinten Nationen) organisiert. Daten werden als Live-Datenstrom aus der ganzen Welt eingesammelt und in aller Regel dateibasiert verteilt (s. Abb. 2).

Ein Standard-DWD-Server verarbeitet ca. 800 GB pro Tag in Form von 300.000 Einzeldateien. Im Server wachsen die Eingangsdaten in internen Caches auf ca. 4 TB Nutzdaten an (v. a. wegen Vorindizierung für schnelle Client-Zugriffe). Aufgrund dieser Datenvolumina stehen Live-Daten nur für ca. 36 Stunden zur Verfügung.

Für ein Softwaresystem wie NinJo stellt alleine die Vielzahl der Eingangsdatentypen und -formate eine große Herausforderung dar, die hochkonfigurative Dateiparser verarbeiten. Komponenten in Client und Server sind „fachliche Säulen“ entlang der Datenstrukturen: Satellitenkomponenten wissen zum Beispiel bewusst nichts von Radarkomponenten, weder in der Softwarearchitektur noch zur Laufzeit. Dies garantiert lose Koppelung und erlaubt eine horizontale Skalierung.

Viele Datenformate

Die WMO bemüht sich, Standards für den weltweiten Datenaustausch festzulegen und stetig weiterzuentwickeln, darunter standardisierte Austauschformate in tabellengesteuerten Formaten.

Die Standards werden periodisch angepasst, sodass mehrere Versionen in Umlauf sind. Nicht alle Lieferanten sind in der Lage, zeitnah auf die neueste Version umzustellen. Bis neue Standards weltweit operativ sind, kann es Jahre oder Jahrzehnte dauern: Ein Messautomat in einem Entwicklungsland kann heutzutage noch Datenformate der 70er Jahre erzeugen. Ein Zusammenspiel moderner JVM-basierter Softwaresysteme wie NinJo mit Fortran77- und C-Programmen ist daher leider der Regelfall. Fehlinterpretationen der Standards oder lokale Eigenarten in der Codierung erfordern oft mühsame und fehleranfällige Konfigurationen beziehungsweise Fallunterscheidungen in der Verarbeitungslogik.

Datenarten	Datenquellen	Datenformate	Datenfrequenzen und -mengen
Bodenmessnetz für Messungen von Temperatur, Druck, Wind, Niederschlag usw.	DWD-eigenes Messnetz mit ca. 1.800 Stationen, weitgehend automatisiert	binäre BUFR-Dateien	Dateneingang im 10-, 30- und 60-Minuten-Takt
	weltweites Messnetz mit über 30.000 Stationen	traditionelle ASCII-Formate und BUFR	4-mal täglich, stündlich, halbstündlich
Radardaten für die Erfassung von Reflektivität, Niederschlag, Wind	DWD-Verbund aus 18 Wetterradaren, dazu Daten aus allen Nachbarländern	binäre BUFR-Dateien	5-Minuten-Takt pro Produkt, entspricht ca. 60 GB/Tag
Satellitendaten (geostationär und polare Umläufer)	Daten von EUMETSAT und über internationale Austauschverträge	georeferenzierte TIFF-Dateien	bis zu 5-Minuten-Takt, ca. 50 GB/Tag in ca. 10.000 Dateien
physikalische Modellrechnungen aus dem Supercomputer erstellen Vorhersagen bis zu 10 Tage in die Zukunft	DWD-Modellkette mit drei Auflösungsstufen (global, europäisch, hochaufgelöst für Deutschland). Der DWD nutzt eine Cray mit ca. 35.000 Rechenkernen und 2x550 TeraFLOPs/s	binäre Grib-Dateien	Modellrechnungen typischerweise alle 3h bzw. alle 6h, insgesamt ca. 10 TB/Tag

Tabelle 1: Hauptdatenarten, die ca. 90 Prozent des täglichen Datenvolumens ausmachen

Hochoptimierte binäre Dateiformate

Seit den 80er Jahren ersetzt die WMO traditionelle Datenformate durch tabellengesteuerte Binärformate [WMO Codes]. Bekannteste Vertreter sind FM92 (GRIB) und FM94 (BUFR). Beide eignen sich für hochoptimierte Datenübertragung und als Speicherformat. GRIB setzt den Fokus auf Daten mit Gitterstruktur, BUFR kann für Punkt- und Flächendaten verwendet werden. Inhalte zeigen in (versionierte) Tabellen und sparen so Speicherplatz ein, zur „Daten-Entschlüsselung" braucht man aber ein passendes Tabellenwerk. Für beide Formate gibt es APIs für sehr viele Programmiersprachen, darunter auch Java.

Historische Altformate finden in der Datenlieferung leider immer noch viel Verwendung. Sie sind in der Regel nicht selbstbeschreibend und führen also nicht ausreichend Metadaten wie Zeitpunkt und Ort einer Messung, Qualitätsflags usw. mit. Meteorologische Softwaresysteme müssen daher statische Metadaten bei sich konfigurieren (Beispiel: Stationskatalog für Name und Geografie der Messstationen).

Wen interessiert schon das Wetter von gestern?

Das Arbeitsplatzsystem NinJo wird kontinuierlich um neue Daten und Produkte erweitert. Es fehlte jedoch lange eine Schnittstelle zum Zentralarchiv des DWD, die es erlaubt, Daten zu entarchivieren, ohne dass Detailkenntnisse über Datenmodell oder über historische Eigenarten nötig sind. Dieses Zentralarchiv umfasst Wetter- und Klimadatenbanken mit ca. 20 PB Daten und ca. 500 TB Cache, der Großteil davon auf ca. 20.000 Bändern mit Roboterzugriff:

- Die meisten Datenarten liegen im Archiv als Rohdaten dateibasiert so ab, wie sie ursprünglich in der Datenquelle erzeugt wurden, also in der gesamten historischen Format- und Dateivielfalt. So müssen beispielsweise Satellitenbilder für die Visualisierung erst von „ihrer Rohform" in ein Rasterformat transformiert werden.
- Eine relationale Datenbank hält darüber hinaus historische Messdaten und -reihen in einem relationalen Schema vor. Manche ihrer Tabellen enthalten mehr als 10^9 Datensätze.

Die Gruppe der Sachverständigen-Gutachter erstellt amtliche Wetter- und Klimagutachten, zum Beispiel für Versicherungsstreitfälle und für Gerichtsverfahren, und benötigt solche historischen Daten. Die Gutachter nutzten bisher ein Altsystem, das im Vergleich zum Zielsystem NinJo viel weniger Datentypen und nur eine vergleichsweise geringe Anzahl von Visualisierungsoptionen offeriert. Nötige Anpassungen des Altsystems wurden weder funktional noch wirtschaftlich als sinnvoll erachtet. Stattdessen galt es, NinJo als *das* zentrale meteorologische Visualisierungswerkzeug im DWD an das Zentralarchiv anzubinden und eine neue Integrationslösung für den Archivzugriff zu schaffen.

Weitere Nutzungsszenarien für historische Daten:

- Anfertigung von Spezialgutachten, z. B. bei Flugunfällen.
- Prüfung meteorologischer Fachverfahren gegen historische Wetterlagen.
- Erstellung von Publikationen über spezielle Wetterereignisse.
- Bewertung der aktuellen Lage im Vergleich zu besonderen Wetterereignissen der Vergangenheit.
- Ausbildung von Meteorologen im DWD-Bildungszentrum.

Umgang mit der Historisierung von Daten, Formaten, Metadaten

Daten liegen im Zentralarchiv mit allen historischen Unzulänglichkeiten, also mit alten Formaten, Tabellen, Codierungen und Metadaten. Wenn ein System wie NinJo diese Altdaten benutzen soll, so erwachsen daraus für Software und Infrastruktur im Vergleich zum Live-Betrieb zusätzliche Herausforderungen. Zwei Lösungsvarianten sind denkbar:

- *Variante a:* Altdaten werden mit „eingefrorenen" (NinJo-) Softwareständen verarbeitet: Wir heben jeden (wesentlichen) Softwarestand auf, der Konfigurationen, Parser und fachliche Komponenten umfasst, wie er zum damaligen Zeitpunkt gültig war. Theoretisch schließt dies Installationen, Bibliotheken, systemnahe Software wie das JRE, bis zum Betriebssystem ein (z. B. durch Virtualisierung).
- *Variante b:* Der aktuelle (NinJo-)Softwarestand verarbeitet soweit möglich auch Altdaten und hebt beispielsweise Parser für historische Formate auf. Metadaten wie ein Stationskatalog müssen historisiert werden, damit zum Beispiel Messdaten an ihrem historisch korrekten Ort angezeigt werden (und Daten nach Stationsumzug nicht falsch positioniert werden).

Wir haben uns für die zweite Variante entschieden, da wir sonst mit zwei wesentlichen Nachteilen umgehen müssten: Eine explosionsartige Zahl von Softwareständen würde ein genaues Nachhalten aller Änderungen und viel organisatorischen Aufwand bei Installation und Betrieb erfordern! Außerdem erfahren alte Softwarestände keine Updates und Bugfixes mehr – das Einfrieren zurrt eben auch alle Fehler und Probleme fest! Manche Altdaten aus dem Archiv müssen vor der Verarbeitung in ein aktuelles Format überführt werden. Manche dieser Transformationen sind einfach, zum Beispiel ein Wechsel auf ein syntaktisch neues, aber semantisch „gleichwertiges" Dateiformat. Die fachliche Komplexität erlaubt jedoch keine vollständige Automatisierung bis ins letzte Detail.

Zugriff auf das Zentralarchiv mit Play und Akka

Wir beschreiben abschließend, wie die neue Archivzugriff-Komponente NAA Daten aus dem Zentralarchiv entarchivieren, nachbearbeiten und an ein System wie NinJo abgeben kann. NAA setzt folgende Java-Frameworks ein:

- Play 2.4: Webdialoge für Einstellung neuer Entarchivierungsaufträge, dazu Statusabfragen.
- Akka 2.3: massiv parallele Auftragsbearbeitung.
- Spring 4.1: Konfiguration eines fachlichen Modulsystems.

Webdialoge mit dem Play-Framework

Unsere Webdialoge basieren auf dem Play-Framework und erlauben den nutzerfreundlichen Archivzugriff. Der Nutzer wählt Datenarten und Wunschzeitraum zur Entarchivierung aus und kontrolliert den Fortschritt: Ein solcher Entarchivierungsauftrag dauert mehrere Minuten bis hin zu Stunden.

Play [Play] ist ein Web-Framework auf Basis von MVC für Java und Scala. Es hat eine skalierbare Architektur, die im JavaSPEKTRUM schon mehrfach vorgestellt wurde [Play-JS11]. Das Play-Framework ist Teil der Reactive Platform von Typesafe/Lightbend und zeichnet sich durch hoch-performante, weil reaktive und asynchrone Abarbeitung der Web-Requests aus: Play benutzt intern Akka und delegiert Web-Requests an dieses Aktorensystem.

Auftragsbearbeitung durch Akka und Spring

Das Aktorenmodell, das Akka zugrunde liegt, stammt aus der IT-Steinzeit: Es wurde von Carl Hewitt bereits 1973 als Modell für nebenläufige Berechnungen definiert. Die Aktoren sind die ausführbaren Einheiten im System. Sie kommunizieren über asynchrones Messaging. Nachrichten werden in Queues gehalten und nacheinander abgearbeitet. Im Modell existiert kein globaler Zustand („share nothing"). Akka ist ebenfalls Teil der Typesafe/Lightbend-Plattform und implementiert dieses Aktorenmodell mit Programmier- und Laufzeitumgebung (Nachrichten, Hierarchie, Fehlereskalation usw.).

Diese Akka-Aktoren sind *der* Kern unseres NAA-Systems und „unsere Arbeitsbienen": Entarchivierungsaufträge werden durch Aktoren abgearbeitet. Sie werden beim Start fachlich vorkonfiguriert und sind in einem Aktorenpool mehrfach instanziiert. Sobald ein neuer Entarchivierungsauftrag eingestellt wird, erzeugt der Webclient eine entsprechende Nachricht an einen Aktor. Eine hochgradig parallele Abarbeitung von Aufträgen durch die Aktoren ist konfigurierbar und sorgt für nötigen Gesamtdurchsatz und effiziente Ausnutzung vorhandener Ressourcen (CPU, RAM).

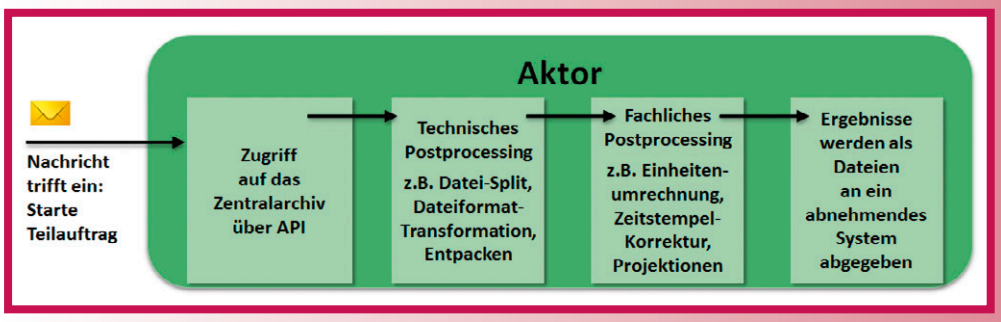

Abb. 3: Aktoren nutzen ein technisches und fachliches Bausteinsystem

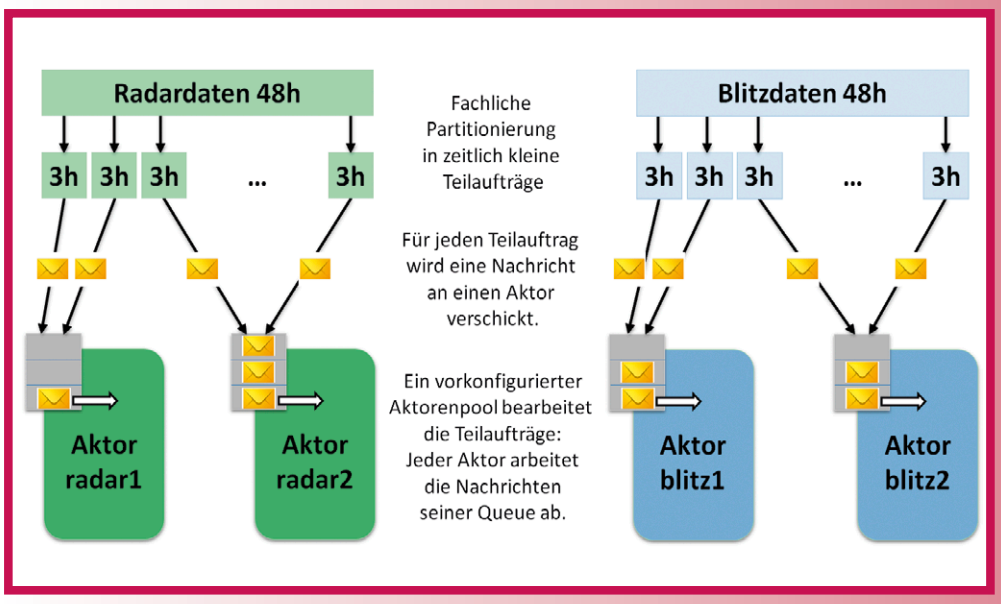

Abb. 4: Fachlich kleinkochen: Parallelisierung mittels Aktoren

Unsere Aktoren basieren auf einem fachlichen Modulsystem, das alle fachlichen Schritte eines Entarchivierungsauftrages festlegt. Diese Einzelschritte nutzen das vordefinierte Bausteinsystem und werden durch das Spring-Framework flexibel zu „fachlichen Aktoren" kombiniert (s. Abb. 3).

Komplex sind vor allem die Postprocessing-Schritte: Das Zentralarchiv speichert Daten in den verschiedensten Formaten ab, was deren nachgelagerte Transformation erfordert. Der Extremfall sind rechenintensive und fachliche Konvertierungen wie z. B. Massendatenverarbeitung von Satellitendaten: Dabei müssen Rohdaten in ein Rasterformat konvertiert, projiziert sowie aufwendig fachlich kalibriert werden, zum Beispiel zur Korrektur von Sonnenstand und Atmosphäre.

Partitionierung als Basis der Parallelisierung

Wesentliche Grundlage der parallelen Auftragsbearbeitung ist dessen fachliche Partitionierung anhand von Datenarten und Abfragezeitraum. Stellt der Nutzer beispielsweise einen Entarchivierungsauftrag für *„Radardaten und Blitzdaten für den 4.+5. März"* ein, so teilen wir diese 48 Stunden in 2x16 kleine Einzelaufträge à 3 Stunden auf. Diese werden an Aktoren im Pool (s. Abb. 4) verteilt.

Die einzelnen Teilpakete sind fachlich unabhängig, auch ihre Reihenfolge spielt keine Rolle. Die Vorteile dieser Partitionierung sind:

- Einfache Parallelisierung und Lastverteilung.
- Monitoring von Teilaufträgen und Statusanzeige (eingestellt, gestartet, laufend, abgeschlossen erfolgreich, abgeschlossen mit Fehler) ist einfach möglich.
- Abbruch und Kontrolle: Wenn ein Aktor eine Nachricht für eine Auftragsbearbeitung erhalten hat, so ist dieser Auftrag de facto „nicht mehr zu stoppen". Holt ein Aktor eine Nachricht aus seiner Queue, so wird sie immer komplett abgearbeitet. Unterbrechbar sind Teilaufträge also nicht. Grund: Die meisten Entarchivierungsschritte sind sehr ressourcenintensiv, zeitaufwendig und technisch blockierend (z. B. wegen blockierender Treiber – daher verwenden wir auch einen separaten Worker-Thread-Pool). Je kleiner Teilaufträge geschnitten sind, desto schneller kommen administrative Nachrichten (Cancel, Suspend usw.) zum Zuge.
- Der Datendurchsatz über die gesamte Kette steigt, wenn fachliche Teilergebnisse schnell vorliegen und bereits an ein nachgelagertes System weitergereicht werden können.

Vordefinierte Akka-Router haben sich für uns als nützlich erwiesen, die Nachrichten an Aktoren effizient durchleiten. Für die parallele Auftragsbearbeitung nutzen wir einen RoundRobin-Pool zur Lastverteilung. Ein ConsistentHashingRouter leitet Nachrichten inhaltsbasiert weiter. Akka stellt weitere Router zur Verfügung, die bekannte EAI-Patterns [Hohpe03] implementieren.

Fazit

Die fachlichen Herausforderungen werden zukünftig nicht kleiner, im Gegenteil: Neue meteorologische Produkte und Verfahren erzeugen immer schneller immer mehr und immer hochaufgelöstere Daten. Erwartete Steigerung der täglichen Datenmenge: bis zu Faktor 10 bis zum Jahre 2025! Unser Archivzugriff bedient gezielt historische Nutzungsszenarien und neue Nutzergruppen. Play und Akka haben sich dabei als performante und stabile Integrationslösung sehr bewährt.

Literatur und Links

[Akka] Akka-Framework, `http://akka.io/`
[HeHa09] D. Heizenreder, S. Haucke, Das meteorologische Visualisierungs- und Produktionssystem NinJo, in: promet, Jahrgang 35, Heft 1-3, 2009
[Hohpe03] G. Hohpe, B. Wolf, Enterprise Integration Patterns: Designing, Building, and Deploying Messaging Solutions, Addison-Wesley, 2003
[Play] Play-Framework, `https://www.playframework.com/`
[Play-JS11] N. Stargardt, Schnelle Webentwicklung in reinem Java mit Play, in: JavaSPEKTRUM, 04/2011
[WMOCodes] WMO International Codes – Migration to tables-driven code forms, `http://www.wmo.int/pages/prog/www/WMOCodes.html`

Martin Lehmann ist Diplom-Informatiker und als Cheftechnologe und Softwarearchitekt bei der Accso – Accelerated Solutions GmbH tätig. Seit Ende der 90er-Jahre arbeitet er in der Softwareentwicklung in diversen Projekten der Individualentwicklung für Kunden verschiedener Branchen.
E-Mail: martin.lehmann@accso.de

Marcus Werner ist Diplom-Mathematiker und arbeitet als Leiter der Gruppe „Softwaretechnische Anwendungsentwicklung“ für den Deutschen Wetterdienst. Seit Ende der 90er-Jahre arbeitet er in der Softwareentwicklung mit den Schwerpunkten Projektmanagement, Anforderungsanalyse und Kommunikation der verschiedensten Stakeholdergruppen mit dem Entwicklungsteam.
E-Mail: marcus.werner@dwd.de