Profil

Dipl.-Inform. Jens Fransson

Senior Softwarearchitekt/-entwickler (Full Stack)

Jens Fransson, Jahrgang 1972, konzipiert und entwickelt seit 1996 selbständig Java-Anwendungen. Besonderes Augenmerk legt er auf Clean Code, SOLID-Prinzipien, Refactoring und Test Driven Development. Er bevorzugt einen funktionalen Programmierstil.

Technische Schwerpunkte

- Java Experte, Erfahrung seit 1996
- Clean Code, SOLID-Prinzipien, funktionaler Programmierstil
- Refactoring, Test-Driven Development
- Continuous Integration & Delivery

Frameworks & Tools

- IntelliJ Idea, Eclipse
- Spring/Spring Boot, Google Guice
- Swing, SWT, Eclipse RCP, Tycho, Vaadin, CUBA, HTML/CSS, Bootstrap, Angular
- Maven, Gradle, Git, Jira, Jenkins, Docker, Heroku, Kubernetes
- Migration von Desktop (Swing, SWT, RCP, ...) auf Web (Angular, JxBrowser, …)

Themen

- Agile, Scrum, Kanban
- Full Stack, Rapid Web Application Development
- Branchen: Logistik, Finanzen, Börse, Industrie, Verlage, Energieversorgung
- Geschäftsanwendungen, Handelssysteme, Quantitative Finance (Quant Developer)

Good programmers write code that humans can understand.

— Martin Fowler in *Refactoring*

Zur Person

Name	Jens Fransson
Ausbildung	Universität, Hamburg Abschluss: Diplom-Informatiker
Berufserfahrung seit	1995
Fremdsprachen	• Deutsch (Muttersprache) • Englisch (fließend) • Französisch (gut) • Latein (Latinum)

Kompetenzen

Schwerpunkte	Softwareanalyse, -design und -implementierung; Code-Reviews, Qualitätssicherung, Clean Code, Refactoring
Spezielle Kenntnisse	Java (seit 1996)

IT-Kenntnisse

Java (Schwerpunkt, Erfahrung seit 1996)	Java-Versionen, JDK 1.0 bis Java 21, Java EE, J2EE, JSP (JavaServer Pages), Servlets, Spring Core, Spring MVC, Spring Security, Spring Social, Hibernate, JPA, AWT, Swing, SWT, JFace, Eclipse RCP, Tycho, JDBC, GWT (Web-GUI), Vaadin (Web-GUI + Backend), CUBA, Struts, Taglibs, JSF, Java Cryptography, GCJ (Gnu Compiler for Java), Excelsior Jet, JAXP, JDOM, MDR (Meta Data Repository), Jakarta Log4J, JUnit, Java Regular Expressions, Java 2D, Piccolo, Jazz (ZGUI Rahmenwerke), JFreeChart, JNI
Weitere Programmier-sprachen	Object Pascal, Borland Delphi, VCL Rahmenwerk, PHP, C/C++, STL, Win32 API, Mumps, Microsoft C#, .NET Rahmenwerk, JavaScript, HTML 5, CSS 3, SACSS, jQuery, Angular
Datenbanken	SQL, Oracle, Postgres, H2, HSQLDB, MariaDB, MySQL, DB2, Borland Database Engine (BDE), MS Access, db4o (Objektorientierte Datenbank)
Markup-Sprachen	XHTML, HTML, CSS, XML, XSL, XSLT (XML Stylesheets), SGML, Bootstrap, Material
Kommunikation	REST, JAX-WS, SOAP, TCP/IP, RMI, JMS (Java Message Service), Serielle Schnittstelle
Entwicklungsumge-bungen	IntelliJ Idea, Eclipse, Netbeans, Borland Delphi, MS Visual Studio .NET, Oxygen

IT-Kenntnisse	
Modellierungswerk-zeuge	Together, Visual Paradigm for UML, Rational Rose, XDE, MagicDraw
Betriebssysteme	Windows 11, 10, 8, 7, Vista, …, 3.1, MS-DOS, Unix (Linux, Solaris, Ubuntu, Red Hat, …)

Branche/Bereich	**Versicherung**
Zeitraum	06/25 – 05/26
Projektziel	• **Altanwendung:** Behebung von Fehlern und Neuentwicklung von Features in einer existierenden Java 17 Jakarta EE Anwendung mit JPA/Hibernate/MS SQL Server Backend und JSF/PrimeFaces Frontend. Die Anwendung ist als modularer Monolith in mehrere Macroservices aufgeteilt. Das Deployment erfolgt auf einem JBoss Server mittels Docker. Datenbankupdates werden mittels Liquibase umgesetzt. Die Anwendung dient zur Abbildung des Kerngeschäftes einer Versicherung. • **Neuanwendung:** Neuentwicklung einer Java 25 Microservices Anwendung zur Abarbeitung von in Java implementierten Workflows (Zustandsautomaten) – eine Art Business Process Engine. Die Anwendung basiert auf Spring Boot 4, JPA, REST (OpenAPI), Artemis, Postgres und Docker. Die Java Workflows werden von einer anderen Anwendung nachrichtenbasiert gesteuert.
Aufgaben/Verantwort-lichkeiten	• **Altanwendung:** Behebung von Fehlern und Neuentwicklung von Features in Front- und Backend (Full Stack). Erstellung von Liquibase Skripten zur Datenbankmigration. Entwicklung von automatisierten Tests in JUnit zur Überprüfung der Änderungen am Code. • **Neuanwendung:** Entwicklung eines Service Provider Interface (SPI) für Workflows in Java. Die SPI erlaubt die Neuentwicklung von Zustandsautomaten, die nachrichtenbasierte Zustandsübergänge ermöglichen und damit den Workflow implementieren. Ein Worker-Service arbeitet einen Workflow ab, indem er Nachrichten empfängt und Zustandsübergänge durchführt. • *Arbeitsprozess:* Aufgabenstellung und Dokumentation der Tätigkeiten erfolgen in Jira im Kanban-Prozess. Die Arbeitsergebnisse werden als Pull Request zum Review in GitLab zur Verfügung gestellt. • IDE: IntelliJ Idea. • Frameworks: Java 17 + 25, JPA, Hibernate, MSSQL, Maven, Git, Gitlab, Docker, Jira, JSF, Spring Boot 4, OpenAPI, REST, Postgres, Artemis

Branche/Bereich	**Öffentlicher Sektor**
Zeitraum	03/24 – 06/25
Projektziel	Performance-Optimierung und Refactoring einer existierenden Eclipse/ Equinox OSGi Anwendung mit JPA/Hibernate/Oracle Backend und Angular Frontend. Die Anwendung (ANBU) dient zur Buchhaltung von Anlagen und wird im Öffentlichen Dienst in mehreren Bundesländern für die Kommunikation zwischen verschiedenen Behörden eingesetzt und basiert auf der Plattform VOIS.
Aufgaben/Verantwortlichkeiten	• Performance-Optimierung einer Datenbankabfrage für einen Angular-Dialog. Erstellung eines Performance/Lasttests zur Überprüfung der Performance. Die Optimierung erfolgt durch das Erstellen einer Projektion mittels Criteria-API. Der Zeitverbrauch im Produktionsbetrieb sinkt durch die Optimierung von mehr als 3 Stunden auf unter 30 Sekunden. • Verbesserung der Paketstruktur, Umzug von Entitäten in andere packages. • Refactoring: Ersetzung von alten Entitäten durch neue. Schaffung und Verwendung von Convertern in Mapstruct zur schrittweisen Migration auf das neue Datenmodell. • Fehlerbehebung im Angular Frontend und Java Backend (Full Stack) • IDE: Eclipse. • Frameworks: Java 17, VOIS, Equinox/OSGi, JPA, Hibernate, Criteria-API, Tycho, Maven, Git, Gitlab, Jira, Docker, Oracle, Angular

Branche/Bereich	**Energiehandel**
Zeitraum	01/24 – 03/24
Projektziel	Neuentwicklung einer Anwendung (Backend/Middleware) zur Verarbeitung von Orderbüchern, Kursdaten und weiteren handelsrelevanten Daten aus dem Energiehandel. Die Anwendung besteht aus mehreren Microservices. Die Daten werden über eine Websocket-Schnittstelle empfangen (Microsoft SignalR/Volue), aggregiert und an einen Kafka-Service (Microsoft Azure Eventhub) weitergeleitet. Für den Fall des Ausfalls der Websocket-Schnittstelle steht ein weiterer Service (Rest-Proxy) bereit, der eine Rest-API (Volue) zum Datenempfang verwendet und dann die Daten an Kafka weiterleitet. Der Ausfall wird automatisch erkannt und durch den Rest-Proxy ersetzt. Für das Monitoring wird Prometheus und Micrometer eingesetzt.
Aufgaben/Verantwortlichkeiten	• Komplette Neuentwicklung der Anwendung und Microservices – Websocket-Proxy, Websocket-Orderbook und Rest-Proxy. Die Entwicklung beinhaltet JUnit- und Integrationstests. In diesem Projekt arbeitet nur ein Entwickler (Jens Fransson). • IDE: IntelliJ IDEA. • Frameworks: Java 21, Spring Boot 3.2.2, Maven, Git, Jira, Bitbucket, Volue, Websockets, Rest, Swagger, Kafka, Microsoft Azure (SignalR, Eventhub), Docker, Prometheus, Micrometer

Projekte/Tätigkeiten	
Branche/Bereich	**Logistik**
Zeitraum	08/18 – 11/23
Projektziel	Weiterentwicklung und Pflege einer Eclipse RCP Anwendung im Logistikumfeld. Die Anwendung dient als Plattform für per OSGi angebundene Kundenmodule (die wiederum eigene Anwendungen darstellen). Sie umfasst ca. 160.000 Codezeilen und läuft auf ca. 40.000 Client-Rechnern. Die Anwendung wird über eine weitere, eigens implementierte Service-Anwendung auf den Client-Rechnern ausgeliefert. Die Anwendung erfüllt die Anforderungen an Kritische Infrastruktur (KRITIS) gemäß der Spezifikation des Bundesamtes für Sicherheit in der Informationstechnik (BSI).
Aufgaben/Verantwortlichkeiten	• Erfassung von Änderungswünschen und Fehlerberichten für die Anwendungen in Atlassian Jira. • Dokumentation der erforderlichen Prozesse im Wiki. Dokumentation und Implementierung gemäß der KRITIS Spezifikation. • Implementation der Anforderungen in Java mit IntelliJ Idea. • Überwachung, Wartung und Weiterentwicklung der Continuous Integration auf Jenkins mit Continuous Integration Pipelines (DevOps). • Unregelmäßige Erstellung und Veröffentlichung von Releases der RCP Anwendung. • Integration der Bibliothek JxBrowser zur Anbindung von Kundenmodulen, die als Web-Anwendung in Angular implementiert sind. • Migration der bestehenden SSO/Kerberos Authentifizierung auf eine MFA-fähige OAuth 2.1/OpenID Implementierung für den Desktop.
Methoden/ Technologien	Java SE 1.8, IntelliJ Idea, Spring 4, Swing, Angular, SWT, Eclipse RCP, OSGi, Maven, Tycho, Nexus, SonarQube, Jira, ClearQuest, Jenkins, Maven, Git/BitBucket, OAuth 2.1, OpenID

Branche/Bereich	**Logistik**
Zeitraum	08/16 – 11/23
Projektziel	Weiterentwicklung einer umfangreichen Java/Spring Swing-Anwendung im Logistikumfeld. Die Anwendung umfasst ca. 3,5 Millionen Codezeilen und läuft auf ca. 40.000 Client-Rechnern. Das Backend läuft auf dynamisch skalierenden Server-Instanzen.

Projekte/Tätigkeiten	
Aufgaben / Verant-wortlichkeiten	• Die Anwendung ist in Unterprojekte gegliedert, für jedes Unterprojekt ist ein Team von ca. 10 Entwicklern zuständig. Insgesamt arbeiten ca. 100 Entwickler in dem Projekt. • Aufgaben: Full Stack – Entwicklung neuer Funktionen, sowohl im Frontend (UI: Swing, Angular) wie auch im Backend (Java Spring Services, JPA, Hibernate, Oracle, Tomcat 7), inklusive der erforderlichen Unit-Tests. Beheben von Fehlern.
Methoden/ Technologien	Java SE 1.8, IntelliJ Idea, Spring 4, Swing, Oracle, Hibernate, Maven, Nexus, Tomcat 7, SonarQube, Jira, ClearQuest, Jenkins, Maven, Git/BitBucket
Branche/Bereich	**Börsenhandel**
Zeitraum	09/15 – 01/21
Projektziel	Entwicklung einer Java-Anwendung zur Simulation von Handelssystemen für Finanzinstrumente (Aktien, Futures, Forex).
Aufgaben/Verantwort-lichkeiten	• Clean Code – Funktionaler Programmierstil (fast alle Objektinstanzen sind unveränderlich) • Durchgängige Einhaltung der SOLID Prinzipien • Umfang der Anwendung: ca. 100.000 Lines of Code, ca. 1.800 Unit Tests. Laufzeit aller Tests ca. 12 sec. • Konfiguration der Anwendung mittels Spring Boot Profile. • Aufgaben: Entwicklung neuer Funktionen inklusive der erforderlichen Unit-Tests.
Methoden/ Technologien	Java SE 17, IntelliJ Idea, Spring Boot, Mockito, Git, Gradle, Maven, TeamCity

Branche/Bereich	**Logistik**
Zeitraum	02/15 – 06/16
Projektziel	Weiterentwicklung einer umfangreichen Java/Java EE Web-Anwendung im Logistikumfeld.
Aufgaben/Verantwort-lichkeite	• Die Anwendung ist in Unterprojekte gegliedert, für jedes Unterprojekt ist ein Team von ca. 5 – 10 Entwicklern, 2 – 5 Requirement Engineers und Business Stakeholdern zuständig. Insgesamt arbeiten ca. 100 Entwickler in dem Projekt. • Aufgaben: Full Stack – Entwicklung neuer Funktionen, sowohl im Frontend (UI: JSF, HTML, CSS) wie auch im Backend (Java EE Services, EJBs, JPA, EclipseLink, Oracle, GlassFish 3), inklusive der erforderlichen Unit-Tests. Beheben von Fehlern. Dokumentation der Implementierung.

Projekte/Tätigkeiten	
Methoden/ Technologien	Java SE 1.7, Java EE 6, JSF, HTML, CSS, Oracle, EclipseLink, GlassFish 3, Google Chrome, SonarQube, ClearQuest, Jenkins, Gradle, Git/Gerrit, IntelliJ Idea, Eclipse

Projekte/Tätigkeiten

Branche/Bereich	**Energieversorger**
Zeitraum	11/12 – 06/14
Projektziel	Weiterentwicklung einer Client/Server Enterprise-Anwendung mit Swing GUI
Aufgaben/Verantwort-lichkeiten	• Die bestehende Enterprise-Anwendung zur Planung und Optimierung von Kraftwerkseinsätzen wird um ein neues Modul zur Planung von Stillständen (Revisionen, Störfälle etc.) erweitert • Hierfür werden neue GUI-Ansichten (GUI-Schicht), Service-Methoden (Geschäftslogik) und Entitäten (Persistenzschicht) entwickelt • Die Entwicklung der Anwendung erfolgt vor Ort in enger Abstimmung mit dem Anwender • Die Unterstützung des Anwenders im laufenden Betrieb gehört ebenfalls zum Aufgabenfeld
Methoden/ Technologien	Java SE 1.6, Swing, Jide, Oracle Weblogic 12 Application-Server, Oracle, H2 SQL, Spring, iBatis, MyBatis Persistenz-Framework, RMI, JMS, Subversion V, Jenkins Build-Server, BoFiT, Gurobi, IntelliJ Idea, Eclipse

Branche/Bereich	**Mobile**
Zeitraum	09/14 – 12/14
Projektziel	Evaluierung der Eignung aktueller Java-basierter Web-Rahmenwerke zur Realisierung einer mobilen Single Page App (SPA)
Aufgaben/Verantwort-lichkeiten	Bei dieser Untersuchung aktueller Java-basierter Web-Rahmenwerke wurden die derzeitigen Möglichkeiten zur Implementierung einer Single Page App in Java bewertet
Methoden/ Technologien	Java SE 1.7, 1.8, GWT, Vaadin, Spring MVC, Spring Security, Spring Social, OAuth2, Gradle, Git, IntelliJ Idea

Projekte / Tätigkeiten	
Branche/Bereich	**Interbankenhandel**
Zeitraum	04/09 – 11/12
Projektziel	Entwicklung und Ausführung eines Handelssystems zur Abwicklung von Devisengeschäften
Aufgaben/Verantwort-lichkeiten	• Ein Prototyp des Handelssystems wird auf der Plattform MultiCharts in der Programmiersprache PowerLanguage entworfen. Auf dieser Plattform findet auch die Überprüfung mittels historischer Kursdaten statt. • Die Validität des Handelsmodells und die fachliche Funktionalität wird im Prototypen sichergestellt. Zur Ausführung von Handelsgeschäften ist MultiCharts jedoch nicht ausreichend. Daher wird das Handelssystem auf der Zielplattform OpenQuant in der Programmiersprache C# neu entwickelt. Die Ausführung der Handelsgeschäfte wird in Echtzeit überwacht. • Zum Abruf der historischen Kursdaten wird ein Java-Programm entwickelt. Die Entwicklung erfolgt mit der Entwicklungsumgebung IntelliJ Idea. Die Kursdaten werden als persistente Beans in der SQL-Datenbank H2 gespeichert. • Zur Vereinfachung des Umganges mit der Datenbank kommt das Apache Commons Framework DBUtils zum Einsatz. Die Daten liegen in sekundengenauer Auflösung vor. Aufgrund der hieraus resultierenden großen Mengen an Daten sind zahlreiche Maßnahmen zur Sicherstellung einer hohen Performance nötig. Das Datenmodell ist besonders platzsparend und effizient ausgelegt. Die Daten werden nicht nur über verschiedene Tabellen verteilt, sondern auch in nach Zeiträumen unterteilte Datenbanken abgelegt.
Methoden/ Technologien	MultiCharts (PowerLanguage), OpenQuant (C#), IntelliJ Idea, H2 SQL, DBUtils (Apache Commons Framework)

Branche/Bereich	**Finanzdienstleistungen**
Zeitraum	10/08 – 03/09
Projektziel	Automatisierter Herstellertest für eine Web-Anwendung
Aufgaben / Verant-wortlichkeiten	• Neuentwicklung von automatisierten Herstellertests mit Java SE 5 für eine J2EE-Web-Anwendung (Plattform: IBM WebSphere, Rapid Application Developer (RAD)). Die Anwendung wird von Banken genutzt und dient der Bonitätsbewertung von Finanzierungen (Rating). • Zur Prüfung der Funktionalität wird das Rahmenwerk HttpUnit verwendet. Es werden Dialogabläufe sowie Feldinhalte und -zustände überprüft. Die Testdaten werden für den Herstellertest in XML angelegt. • Außerdem findet noch eine weitere Form des Tests statt, in dem die Ergebnisse von Berechnungen der Anwendung mit vom Kunden gelieferten Testdaten abgeglichen werden. Hierbei werden die Testdaten über eine Microsoft Access-Datenbank eingelesen und automatisch gegen die Rechenergebnisse der GUI geprüft.
Methoden/ Technologien	Java SE 5, HttpUnit, JUnit, DBUnit, JavaDoc, Mercury Quality Center, Log4J, Apache Commons, Microsoft Access, Eclipse, Oxygen, CVS, Ant, Wiki

Branche/Bereich	**Logistik**
Zeitraum	02/08 – 08/08
Projektziel	Architektur & Integration für eine Logistikanwendung
Aufgaben/Verant-wort-lichkeiten	• Entwicklung einer Komponente zur Messung der Performance einer performancekritischen Logistikanwendung für Containerhäfen. Die Anwendung ist stark heterogen und verteilt. Die Kommunikation erfolgt über JMS (Java Message Service). • Zur Überwachung der Performance werden die JMS-Nachrichten zusammen mit ihren Sendezeiten mittels JPA (Java Persistence API) und Hibernate in einer Oracle-Datenbank gespeichert. Die primär eingesetzten Programmiersprachen sind Java 6 und C#. Das Projekt wird mit Maven 2 gebaut, geringfügig werden Ant-Skripte eingesetzt. • Das Projekt ist in Komponenten (Bundles, Services) im Sinne einer OSGi-Anwendung aufgeteilt. Die Komponenten werden mit Spring konfiguriert. • Die Builds werden mit CruiseControl gesteuert. Programmteile werden aus mit MagicDraw 15 erstellten UML-Modellen in Verbindung mit OpenArchitectureWare 4 generiert. Die OSGi-Komponenten werden mit Spring konfiguriert.
Methoden/ Technologien	Maven 2, Java SE 6, MagicDraw 15, OpenArchitectureWare 4, Spring, Oracle, Hibernate, JPA Java Persistence API, JMS Java Message Service, Subversion, JUnit, Log4J, Apache Commons, IntelliJ Idea 7, Eclipse 3.4, CruiseControl, Ant, Jira, Test Track Pro