eitco

eitco PROSUITE

Die offene Anwendungsentwicklungsplattform der EITCO

WHITEPAPER

Inhalt

1 Einleitung

1.1 Marktentwicklung

Die 1990er und 2000er Jahre waren geprägt von einem Trend hin zu Standardapplikationen. Die zu dieser Zeit zur Verfügung stehenden Technologien ließen unter wirtschaftlichen Gesichtspunkten individuelle maßgeschneiderte Lösungen für Unternehmen oft nicht zu.

Heute wird allerdings die Passgenauigkeit von Software quer durch alle Branchen für Unternehmen und Einrichtungen immer mehr zum Erfolgs- bzw. Wettbewerbsfaktor. Nahezu für alle Produkte und Dienstleistungen und durch alle Branchen hindurch.

Um diesen Wettbewerbsvorteil zu nutzen, ist es mehr denn je erforderlich, Applikationen genau den Anforderungen und Bedürfnissen der Unternehmen entsprechend zu produzieren. Man kann in diesem Sinne zwischen einem „Standard-Backbone-System“ (klassische ERP-Komponenten mit angemessener Standardisierung) und individuellen Satelliten unterscheiden (die häufig von den großen ERP-Herstellern mitadressiert werden, aber bei mangelnder Flexibilität oft keinen adäquaten Nutzen bieten). Signifikante Wettbewerbsvorteile werden nicht mehr durch klassische ERP-Komponenten erzielt, sondern durch flexible Applikationen, die schnell und effektiv den Zugang zu Kunden managen oder neue Abläufe in schlanker Form einfach und günstig organisieren.

Im Zusammenhang mit Micro-Services und einem sehr schlanken und pragmatischen Service-Bus-Verfahren (als Schnittstellen und Migrationstechnologie) erleben wir eine signifikante Vereinfachung der Integration neuer Anwendungen in bestehende IT-Landschaften. Das führt zu einer Renaissance von Individualentwicklungen in Symbiose zu traditionellen ERP Backbone-Systemen.

In jüngster Vergangenheit haben gerade Java-Technologien für die Entwicklung von Unternehmens-anwendungen eine signifikante Bedeutung gewonnen. Mit der Java-Plattform Enterprise Edition (JavaEE) und Standard Edition (JavaSE) steht eine Software-Architekturbasis zur Verfügung, das verteilte, mehrschichtige Anwendungen zulässt und in jüngster Zeit immer effizienter und pragmatischer wurde.

Heute spielen bei der Java Software-Entwicklung Open-Source Technologien eine sehr bedeutende Rolle. Quelloffener Produkte in der Java-Software-Entwicklung sind weit verbreitet. Dass Open-Source Software auch in den deutschen Unternehmen angekommen ist, wird durch viele Studien belegt.

Allerdings gibt es auch Meinungen im Markt, die die Unübersichtlichkeit, die Nutzungsmöglichkeit und die zu stark einem Trend folgende (Hype-)Produktion im Open-Source Umfeld bemängeln. Viele Technologien werden zu früh eingesetzt und sind bei der Nutzung kontraproduktiv.

Daher ist unter anderem die Prüfung der Technologien auf Reifegrad, Sicherheit und Zukunftsperspektive von größter Bedeutung, bevor diese eingesetzt werden.

Neben der Herausforderung, eine Technologie bzw. eine Komponente zu finden, die die gestellten Anforderungen erfüllt, gilt es auch, die gewünschte Integration einer solchen Komponente im geplanten Einsatzumfeld sicherzustellen. Es ist notwendig, herauszufinden ob der betreffende Baustein stabil und verlässlich ist. Denn in der Anwendungsentwicklung gilt – wie in anderen Systemen auch -, dass die Qualität des Ganzen stets durch das schwächste Glied in der Kette bestimmt wird. Insofern besteht bei der Planung von Systemen die große Herausforderung in der

optimalen Zusammenstellung von Bausteinen, die sich am Ende auf das Verhältnis zwischen Quantität, Qualität und Aufwand niederschlägt.

1.2 Eine offene Plattform für individuelle Applikationen

Anfang 2008 startete die European IT Consultancy EITCO GmbH (EITCO) mit der Kompetenz aus über 20 Jahren Softwareengineering den Entwurf einer offenen Java-Anwendungsplattform, mit der hoch effiziente, individuelle Applikationen entwickelt werden können.

Folgende Kriterien wurden dabei besonders hoch gewichtet:

1. Wirtschaftlichkeit
   1.1. hoher Grad der Wiederverwendung einmal implementierter Geschäftslogiken
   1.2. kurze, planbare Entwicklungszeiten
   1.3. kurze Einarbeitung von Mitarbeitern bzw. projektbezogene Integration von bestehenden Mitarbeiterteams

2. Personalsicherheit
   2.1. Auswahl an Technologien hinsichtlich Verständlichkeit
   2.2. Auswahl an Technologien hinsichtlich Mitarbeitermotivation (State-Of-The-Art-Technologien)

3. Betriebskriterien
   3.1. IT-Sicherheit
   3.2. Betriebssicherheit
   3.3. Wartbarkeit

4. Offenheit
   4.1. API
   4.2. Unabhängigkeit zu Endgeräten und Präsentationsformen
   4.3. Flexible Unterstützung von Vorgängen, Abläufen bzw. Workflows
   4.4. Konnektivität zu Drittsystemen sowohl als Anbieter als auch Nutzer

Aus der Erfahrung zeigt sich schnell die Gefahr, die in der Vorstellung immer neuer Frameworks und "Moden" liegt. Industrielle Softwareentwicklung ist nicht durch das Aufspringen auf immer neue Trends möglich. Leitlinie der Komposition industriell tragfähiger Basistechnologien ist die Reduktion auf bewährte, möglichst wenig komplexe und wirklich nützliche Fortschritte. Nicht der Spieltrieb und schnell realisierbare "Gimmicks" sind entscheidend. Dabei muss durch Schichtung und Lokalität der Komponenten die Plattform robust gegenüber dem Austausch von Basistechnologien sein. Jede Entscheidung muss geplant und mit überschaubarem Aufwand revidierbar sein. Nur dann erhält man ein Grundgerüst, in das man solide Entwicklungen der „Open-Source-Szene“ professionell integrieren und somit von ihnen profitieren kann. Dies gilt umso mehr, da die Entwicklungen in hoch innovativen "Biotopen" wie JavaEE/SE sehr dynamisch sind und auch immer wieder zu konzeptionellen Revisionen führen. Diese "radikale Fortschrittlichkeit" muss an die Leine genommen werden.

Unter der Voraussetzung professioneller Verwendung erhält der Kunde jedoch dramatische Kosten- und Qualitätsvorteile, die zudem noch durch eine relative Herstellerunabhängigkeit gekennzeichnet sind.

2 ProSuite

2.1 Grundsätze

ProSuite bezeichnet zum einen ein technisches Raster (technische Architektur), in dem offene Basistechnologien auf effektive und effiziente Art und Weise zusammen spielen und die im Wesentlichen der JavaEE/SE (Java Enterprise/Standard Edition) folgen. Zum anderen bezeichnet ProSuite eine stetig wachsende Sammlung sehr generischer fachlicher Komponenten (fachliche Architektur), die bei der Entwicklung von Applikationen als Fertigbauteile oder Vorfabrikate direkt verwendbar sind und die Produktivität und Planbarkeit stark erhöhen.

Ein Grundsatz von ProSuite ist es daher, eine Anwendungsplattform mit **hoher technischer als auch fachlicher Vorfertigung** bereitzustellen.

Dabei bildet die technische Vorfertigung stets die Grundlage für eine effiziente Anwendungsentwicklung, die fachliche Vorfertigung soll jedoch im Fokus stehen. Dieses entscheidende Alleinstellungsmerkmal zieht sich durch alle Ebenen von ProSuite und bedeutet nichts anderes als das Primat der konkreten Kundenanforderung im konkreten Projekt. Hierauf ist auch der gesamte Produktionsprozess einschließlich der Methoden der geschäftsprozessorientierten Anforderungserhebung und deren Umsetzung mit ProSuite fokussiert. Sämtliche technischen Artefakte bleiben im Grundsatz austauschbar und sind nur Mittel zum Zweck. Durch die vorgedachte Austauschbarkeit der Mittel ist die Plattform und damit die Kundenanwendung gegen technische Überalterung sehr gut geschützt. Anstelle von schmerzlichen „Big-Bang-Ablösungsszenarien“ werden in einer langfristigen Rückschau mit ProSuite immer sanfte und kostengünstige Evolutionspfade erkennbar sein, die die Investitionen des Kunden geschützt halten.

Daraus folgend gilt für ProSuite implizit auch der Grundsatz der **bestmöglichen fachlichen Wiederverwendbarkeit.**

Die Architektur von ProSuite gestaltet sich somit derart, dass einerseits die Austauschbarkeit von Techniken und andererseits auch eine hohe Wiederverwendbarkeit von Artefakten gegeben sind. Bezogen auf Wiederverwendbarkeit von Artefakten wird ein besonderes Augenmerk auf eine sehr leistungsfähige Komponentenarchitektur und Komponentenkapselung gerichtet.

Mit ProSuite sollen keine aufwendigen und proprietären Neuerfindungen aufgesetzt werden, sondern es gilt stets, den Ansatz der klugen Assemblierung von bewährten Werkzeugen, Frameworks und Artefakten zu verfolgen.

2.2 Wesentliche Eigenschaften

ProSuite zeichnet sich neben der technischen und fachlichen Vorfertigung mit allgemein hoher Wiederverwendung durch folgende konkrete Eigenschaften aus:

1. Komponentenarchitektur
2. Multi-Channeling
3. Prozessteuerung

Im Vordergrund steht dabei stets die Einhaltung des Prinzips „**keep it simple“** und die **Konzentration auf das Wesentliche**. Es geht nicht darum, mit „Hypes“ zu brillieren, sondern anhand bewährter Technologien und Werkzeugen im Sinne von Best Practices hohe Produktivität und Nachhaltigkeit zu schaffen.

Komponentenarchitektur

Unabhängig von konkreten Technologien gilt allgemein in Bezug auf Architektur und Modularität, dass sich leistungsfähige Plattformen beziehungsweise Anwendungssysteme neben der durchgehenden Einhaltung offener Standards auch durch das Prinzip der modularen, komponentenorientierten Gestaltung auszeichnen.

Die Einhaltung von Standards bildet die Basis für die unabdingbaren Eigenschaften **Verständlichkeit** und **Wartbarkeit**. Das Prinzip der modularen Gestaltung dient dazu, die Wartbarkeit und Erweiterbarkeit des Systems derart zu unterstützen, dass Module beziehungsweise Komponenten entsprechend einfach entfernt, ausgetauscht und hinzugefügt werden können. Eine klug geplante Komponentenarchitektur bildet außerdem den Grundstein für eine systemübergreifende Wiederverwendbarkeit.

So werden Zugriffe auf externe Komponenten in der ProSuite-Architektur von eigenen ProSuite-Hüllen ummantelt. Über diese Hüllen werden Aufrufe und Interaktionen zu/mit anderen Komponenten geregelt. Durch diese Ummantelung ist es ein Einfaches, bei Bedarf eine solche Komponente durch eine andere zu ersetzten. Vor allem die Wartungsgeschwindigkeit kann hierdurch erhöht und die Fehleranfälligkeit drastisch gesenkt werden.

Eine unter diesen Aspekten verstandene Komponentenarchitektur bildet die technische Leitlinie bezogen auf Planung und Umsetzung von ProSuite. Sie genügt höchsten Ansprüchen industrieller Software-Produktion.

Multi-Channeling

In ProSuite ist durch die Trennung in vertikale Schichten eine klare technische Verantwortlichkeit gegeben. Die Business-Logik ist von der Anwendungspräsentation abgegrenzt. Über eine spezielle Schicht (Device Independent Service Layer) wird die Verbindung zwischen den unabhängigen Ausgabegeräten vorgenommen.

Ziel ist es, dass bei der Wahl der Endgeräte und der grafischen Benutzungsoberflächen (GUI; Graphical User Interface) die Anforderungen der Projekte und nicht eine starre Vorgabe von ProSuite in Vordergrund stehen.

ProSuite bringt von Haus aus im Sinne der Vorfertigung ein eigenes GUI mit. Dieses kann im Rahmen der Vorfertigung im Projekt verwendet werden. Gleichermaßen kann ein projektbezogenes GUI Verwendung finden. Dies ist vor allem dort anzuraten, wo eine neue Applikation in eine mit einem durchgehenden GUI bestehende IT-Landschaft integriert werden soll.

Entsprechend dem GUI soll auch die Wahl der Endgeräte und der Hardwarearchitektur durch das Kundenprojekt und nicht durch die Entwicklungsplattform bestimmt werden. Dies ist umso wichtiger, da die Kontakt-/Vertriebskanäle der Kunden ganz wesentlich durch neue Technologien auf der Seite der Benutzerinteraktion beeinflusst werden (Mobile Devices; Social Networks; Speech etc.). Diese neuen Kontakt- und Vertriebskanäle müssen bisweilen explorativ und über „trial and error“- Prozesse

vom Management erschlossen/bewertet werden und dürfen nicht zu jeweils unkalkulierbaren IT-Projekten mutieren. Sie müssen schnell verfügbar sein und bestehende Geschäftslogiken unkompliziert in „neuen Kleidern“ verfügbar machen.

Mit ProSuite **Multi-Channeling** ist genau das sehr einfach möglich: die Bereitstellung der gleichen Anwendung auf verschiedenen Ausgabegeräten bzw. Ausgabekanälen. Je nach Anforderung können Geräte wie

1. Rich-Client
2. Thin-Client
3. Web-Client
4. Mobile-Client
   4.1. Smart-Phone
   4.2. iPhone, iPad
   4.3. Industrie-Endgeräte
5. Voice-Client

sehr effektiv und effizient bedient werden. Dadurch wird die Kreativität des Kunden bei der Umsetzung von Geschäftsmodellen und der Nutzung von Kundenzugängen nicht durch die Anwendungsentwicklung verzögert oder wirtschaftlich verhindert. Die Erstellung entsprechender Anwendungen erfolgt auch nicht in einer "Quick & Dirty" Manier, sondern Geschwindigkeit, industrielle Anwendungsqualität und Nachhaltigkeit der Investitionen werden bei ProSuite miteinander verbunden.

Abbildung 1: Layout der ProSuite-GUI für iPhone

Vorgangs- und Workflowsteuerung

Um den Anforderungen einer Vorgangssteuerung, eines Genehmigungsverfahrens oder allgemein einer Workflowsteuerung gerecht zu werden, enthält ProSuite eine dynamische Steuerung der Anwendungslogik.

Insofern gilt es, Techniken bereitzustellen, die es erlauben, bestimmte Änderungen der Anwendungslogik auf möglichst einfache Art und Weise und auch möglichst innerhalb der Laufzeitumgebung durchführen zu können. Hier kommen die so genannten **Prozess-Engines** zum Tragen. Prozess-Engines unterstützen unter anderem solche Umsetzungen derart, dass keine aufwendigen Build- und Deploy-Prozesse mehr notwendig sind.

Mit ProSuite ist eine Umgebung vorhanden, welche dieses Grundgerüst bietet, um definierte Vorgänge und Workflows in Applikationen abzubilden. Dafür steht ein graphisch unterstützter Workflow-Designer zur Verfügung. Mit diesem Designer können klassische State-Event-Modelle umgesetzt werden, welche dann von der ProSuite-Workflow-Engine ausgeführt werden.

Somit können in ProSuite spezifische Workflows nicht nur sehr einfach erstellt, sondern auch innerhalb der Laufzeitumgebung gepflegt werden.

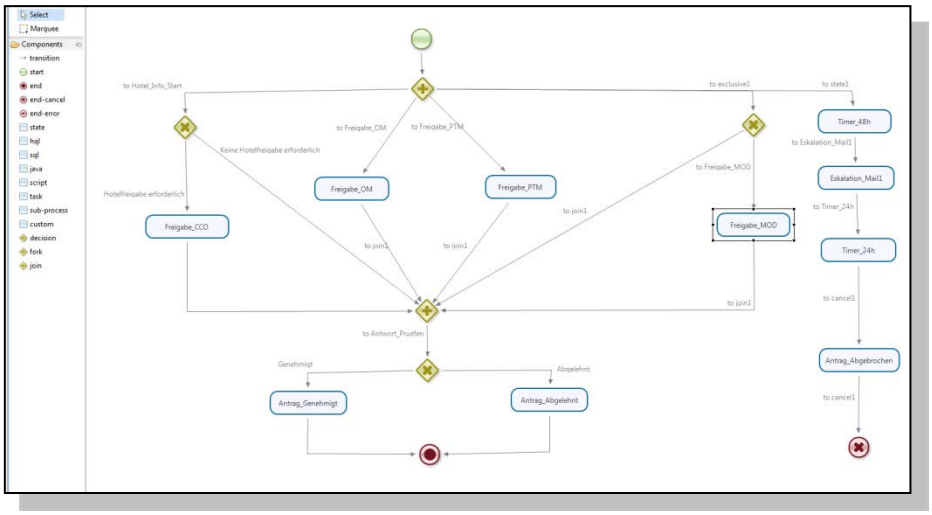

Abbildung 2: Grafischer Workflow-Designer mit Komponenten Repository

3 Technische Basis

ProSuite vereint unterschiedliche Technologien zu einer gemeinsamen Basis, um vielfältige Anwendungsanforderungen umsetzen zu können. Dazu werden standardisierte Komponenten eingesetzt, um Lösungen für häufig wiederkehrende Anwendungsfälle und Anforderungen bereitzustellen.

Darüber hinaus zählt die Integration frei verfügbarer Frameworks und Bibliotheken, die standardisierte Technologien implementieren, zu einer der Hauptaufgaben von ProSuite. Die Quelle für solche Integrationen bildet in erster Linie Open Source, das zahlreiche potentielle Integrationskomponenten zur Verfügung stellt.

ProSuite basiert zu 100% auf der plattformunabhängigen Programmiersprache Java. Damit wird eine Flexibilität bei der Auswahl der zu unterstützenden Betriebssysteme gewährleistet. Mit ProSuite entwickelte Anwendungen richten sich nach dem Client-Server-Modell basierend auf einer Mehrschichtarchitekur. Anwendungen werden als JavaEE/SE-Anwendungen mittels eines JavaEE/SE-Servers bereitgestellt.

3.1 Verwendete Frameworks und Standards

Derzeit integriert und nutzt ProSuite u.a. die folgenden Technologien, Frameworks und Standards:

- JavaSE-/JavaEE-Unterstützung
  - JavaSE-Server Apache Tomcat als Anwendungscontainer
  - JavaEE-Server JBoss als Basisplattform. Andere JavaEE konforme Server möglich

- JMX
  - Verwaltung der Service-Komponenten; Impementierung als JMX-Standard-MBeans
  - http://www.oracle.com

- JAX-WS
  - Erzeugung von Webservices via SOAP
  - http://www.jcp.org

- JAX-RS
  - Erzeugung von Webservices auf Basis von REST
  - Nutzung JAX-RS-Referenzimplementierung Jersey
  - http://www.jcp.org

- JPA und Hibernate
  - Verwendung als Object-Relation-Mapper
  - http://www.hibernate.org

- JAAS im Zusammenspiel mit der ProSuite-Security-Komponente
  - Authentifizierung und Autorisierung von Zugriffsrechten
  - Nutzung der Spring-Security-Komponente - http://static.springsource.org

- SmartGWT oder Vaadin
  - Standardframework für einen Browser-Client; andere GUI-Frameworks sind möglich
  - https://vaadin.com/framework

- Activity
  - Prozess-Engine
  - http://www.activiti.org

- BIRT
  - Reporting-Tool
  - http://www.eclipse.org/birt

- Eclipse
  - Modellgetriebene Softwareentwicklung
  - http://www.eclipse.org

- Jenkins
  - Continuous Integration
  - https://jenkins.io

- Apache Maven
  - Build Tool
  - http://maven.apache.org/

- Junit
  - Unit Tests
  - http://www.junit.org

- Apache Tomcat
  - Servlet Container
  - http://tomcat.apache.org

3.2 Architektur

Die folgende Abbildung skizziert die ProSuite-Mehrschichtarchitektur, bestehend aus den vertikalen Schichten:

1. Presentation-Layer
2. Business-Logic-Layer
3. Service-Layer
4. Data-Access-Layer

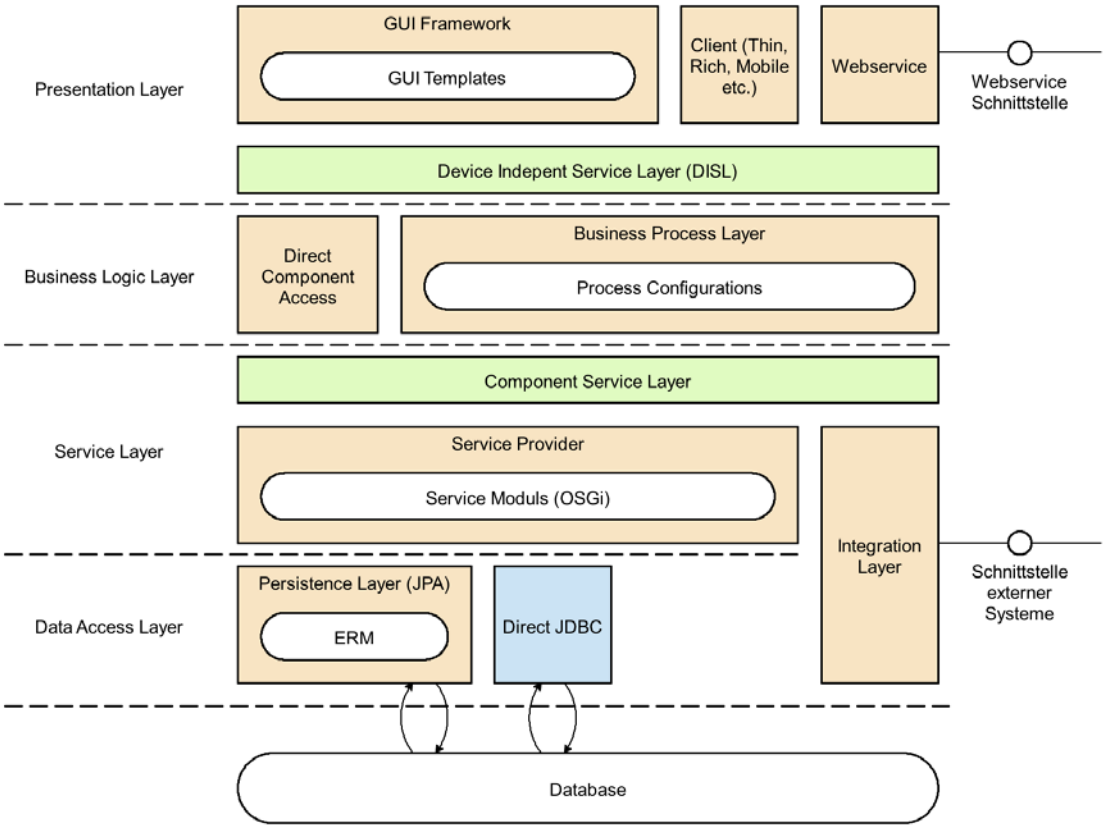

Abbildung 3: ProSuite Mehrschichtarchitektur

Vertikale Schichten

Die Trennung in vertikale Schichten stellt in erster Linie die Delegierung der technischen Verantwortlichkeiten innerhalb des Systems dar. Die Kommunikation zwischen den Schichten findet ausschließlich zwischen zwei benachbarten Schichten statt. In horizontaler Richtung ist ebenfalls eine Trennung im Sinne einer Modularisierung gegeben. Wobei Module nach technischen und auch nach fachlichen Gesichtspunkten differenziert werden. Das Schichtenmodell spiegelt nicht nur die Strukturierung der Laufzeitumgebung wider, sondern auch die der Entwicklungsumgebung hinsichtlich Rollen, Prozesse und Tools.

Die Trennung nach Schichten und damit auch im weiteren Sinne nach Komponenten bietet, wie einleitend erwähnt, viele Vorteile bezogen auf die Eigenschaften Flexibilität und Wartbarkeit. Ein weiterer wichtiger Aspekt ist die **Skalierungsfähigkeit** eines Systems, die im Falle der ProSuite-Architektur in hohem Maße gegeben ist: Die Schichten lassen sich horizontal über verschiedene Hardware-Systeme hinweg optimal skalieren. Eine komponentenorientierte Sicht innerhalb der jeweiligen Schicht führt außerdem zu einem Baukastensystem und somit bezogen auf die ProSuite-Produktlinien zu einem bestmöglichen Grad an Wiederverwendbarkeit der Komponenten.

Den Entwicklungsaspekt betrachtend, bietet die Mehrschichtarchitektur die Möglichkeit, Entwicklungsteams zu bilden, die unabhängig von der zu erstellenden Gesamtapplikation an einzelnen Layern arbeiten und mit wohldefinierten Schnittstellen entwickeln können.

Presentation-Layer

Der Presentation-Layer stellt Eingabe und Ausgabe beziehungsweise die Visualisierung von Daten zur Verfügung. Durch das Konzept des **Device Independent Service Layers (DISL)** wird der Zugriff auf Prozesse und Services gekapselt.

Die Nutzer des DISL sind die so genannten DISL-Clients, bei denen es sich in der Regel um GUI-Komponenten handelt. Weitere DISL-Clients sind beispielsweise Voice-Clients oder generell Drittsysteme. Der DISL bildet somit die Basis des Multi-Channeling-Ansatzes der ProSuite. Bezogen auf GUI-Komponenten können beliebige DISL-Clients in Form von spezifischen GUI-Frameworks (Thin-, Rich-Clients etc.) angeschlossen werden.

Dabei wird die Leitlinie der klugen Assemblierung verfolgt. D.h. vorhandene und in Frage kommende GUI-Frameworks werden in ProSuite integriert.

Aktuell nutzt ProSuite als GUI-Client einen Web-Client, basierend auf SmartGWT oder Vaadin.

Webservices zur Servicebereitstellung für Drittsysteme stehen ebenfalls zur Verfügung. Diese werden genauso als DISL-Clients implementiert.

Außerdem ist die Bereitstellung von GUI-Artefakten in Form von Portlets (JSR 286) und die Anbindung von Mobile-Applikationen möglich.

Der ideale Entwicklungsprozess eines DISL-Clients ist abhängig von der verwendeten Technologie und deren Integrationsmöglichkeiten. Bei Bedarf wird die EITCO für die einzelnen Technologien jeweils ein projektspezifisches Templating-System erstellen, um die GUI-Entwicklung zu beschleunigen.

Business-Logic-Layer

Der Business-Logic-Layer enthält die primäre Applikationslogik. Er bietet die Orchestrierung der Services des Service-Layers in Form von domänenspezifischen Prozessen. Der Business-Process-Layer kapselt eine Prozess-Engine, die in der Lage ist, vordefinierte Prozesse auszuführen. Diese Prozesse können sowohl technischer als auch fachlicher Natur sein.

Dabei können im ProSuite-Umfeld synchrone Client-Request-Verarbeitungen oder zeitlich gesteuerte Abläufe, wie beispielsweise Reporterstellungen, ebenso als Prozess definiert werden, wie z.B. komplexe Commerce-Workflows mit Benutzerinteraktionen und Zugriff auf Drittsysteme (z.B. Kreditkarten-Clearing).

Um diese Aufgaben erfüllen zu können, bietet die Prozessspezifikation einfache Logikoperatoren wie Switches und Loops. Fehler und Ausnahmebehandlung sind ebenfalls in der Prozessdefinition enthalten.

Die Prozess-Sicht ermöglicht die Applikationslogik per Konfiguration zu steuern, ohne die dazu notwendigen Softwareartefakte anpassen zu müssen. Gleichzeitig sind die Prozesse implizit dokumentiert und entsprechend leicht verständlich für Personen mit unterschiedlichem technischen

und fachlichen Hintergrund. Der graphische Workflow-Designer ermöglicht das Abbilden der definierten Vorgänge und Workflows. Diese durchlaufen die gleiche Qualitätssicherung wie die herkömmlichen Softwareartefakte und werden nach einem definierten Prozess freigegeben.

Da bestimmte Anwendungsfälle nicht unbedingt eine Prozessdefinition erfordern, ist die direkte Inanspruchnahme von Diensten des Service-Layers durch den Presentation-Layer erlaubt.

Service-Layer

Der Service-Layer stellt mit einer Vielzahl von Diensten das Baukastensystem der ProSuite dar. Die Dienste sind als Service-Module nach dem JMX-Standard implementiert, melden sich selbständig an der zentralen ProSuite-Registry an, und aktualisieren fortlaufend ihren Status. Im ProSuite-Kontext werden die Dienste auch allgemein als Komponenten bezeichnet, die in

1. anwendungspezifische Komponenten
2. produktlinienspezifische Komponenten
3. übergreifende fachliche Komponenten (z.B. Lexikon-/Verzeichniskomponenten) und
4. übergreifende technische Komponenten (z.B. Protokollierungskomponente)

klassifiziert werden. Die klassifizierten und auch katalogisierten Komponenten werden system-übergreifend anhand entsprechender Repositories bereitgestellt.

Die Komponenten haben für datenorientierte Operationen Zugriff auf den Data-Access-Layer und werden über den Component-Service-Layer der darüberliegenden Schicht bereitgestellt. Die Komponenten haben über den Component-Service-Layer auch selbst Zugriff auf andere Komponenten. Der Service-Layer kapselt über den Service-Provider die konkrete JMX-Schnittstelle, sodass ein transparenter Zugriff auf die Komponenten gewährleistet ist und dadurch die konkreten Komponentenimplementierungen austauschbar sind. Der vertikale Integration-Layer bietet innerhalb des Service-Layers den Zugriff auf externe Datenquellen über den Data-Access-Layer und auch allgemein den Zugriff auf Drittsysteme.

Die Komponenten werden nach klassischen Java-Entwicklungsansätzen mit IDEs (aktuell Eclipse) programmiert. Der Komponentencontainer, gegen den entwickelt wird, ist durch den JMX-Standard definiert. Automatisierte JUnit-Tests und die Verwendung einer Mock-Serviceumgebung sichern die Qualität der Implementierung.

Data-Access-Layer

Der Data-Access-Layer stellt die Verbindung zum DBMS her und abstrahiert SQL-Befehle sowie datenbankspezifische Syntax gegenüber den Komponenten. Auf Basis der Java Persistence API (JPA) werden Domainobjekte zur Verfügung gestellt, die erzeugt, gelesen, aktualisiert und gelöscht werden können. Domain-Objekte stehen über ein Entity-Relationship-Model (ERM) in Beziehung. Diese Beziehungen manifestieren sich als Relationstabellen in der Datenbank. Als konkrete JPA-Implementierung wird Hibernate verwendet. Weitere JPA-konforme Frameworks sind ebenfalls nutzbar. Außerdem enthält der Data-Access-Layer Module aus dem vertikalen Integration-Layer, der einen transparenten Zugriff auf externe Datenquellen, beispielsweise über Webservices, zur Verfügung stellt.

Für einen direkten Datenbankzugriff via JDBC stellt der Data-Acces-Layer ebenfalls entsprechende Funktionalitäten zur Verfügung.

In der ProSuite-Produktion sind MySQL und Oracle als Standards gesetzt, auf Wunsch des Auftraggebers können auch andere Datenbankmanagementsysteme (DBMS) qualitätsgesichert eingebunden werden. Durch Verwendung von JPA wird eine weitgehende datenbankunabhängigkeit hergestellt.

Der Entwicklungsprozess im Sinne der Datenstrukturmodellierung kann durch UML- beziehungsweise konkrete ERM-Modellierwerkzeuge unterstützt werden.

4 ProSuite Komponenten

In Bezug auf die fachliche Vorfertigung stellt ProSuite eine stetig wachsende Basis fachlicher Komponenten zur Verfügung. Die folgende Liste fachlicher Komponenten soll eine Vorstellung von der Granularitätsebene vermitteln, auf welcher vorgefertigte fachliche Komponenten in ProSuite in der Regel angesiedelt sind:

1. Startkomponente
2. Terminmanagement
3. Antragsmanagement
4. Antragsmanagement – Designer
5. Genehmigungsmanagement
6. Genehmigungsmanagement – Designer
7. Messaging, z.B. E-Mail
8. Reportmanagement
9. Reportmanagement – Designer

In einem speziellen Auditing Prozess, bestehend aus technischen und fachlichen Spezialisten der EITCO, wird regelmäßig beurteilt, welche Komponenten aus Projekten möglicherweise als generische Kernkomponenten im Sinne von vorgefertigten Bausteinen qualifiziert werden können. Im Falle positiver Evaluierung erfolgt eine (weitere) Generalisierung und Aufnahme in das Repository der Vorfabrikate.

5 Merkmale der ProSuite-Entwicklung

Die Konzeption und Entwicklung von ProSuite wurde mit der jahrelangen Kompetenz aus einer Vielzahl von IT-Projekten und modernsten Methoden der Entwicklung kombiniert. Weitere Erfolgsfaktoren waren und sind:

1. Spezielles externes Know-how wurde in der frühen Phase der Entwicklung mit einbezogen
2. Erfahrungen von Dritten wurden in der Konzeption berücksichtigt, um „aus den Fehlern anderer zu lernen“
3. Qualitätssicherung wird groß geschrieben:
   - 3.1. Zertifizierung des Entwicklungsprozesses nach ISO 9001:2008
   - 3.2. Klare Definition der Anforderungen und Erstellung des ProSuite Lastenheftes mit Arcway Cockpit (www.arcway.com)
   - 3.3. Qualitätsgesicherter Entwicklungsprozess gemäß V-Modell mit automatischer Übernahme des Lastenheftes in die Qualitätskontrolle
   - 3.4. Testautomation
4. Aufbau eines kompetenten Entwicklungsteams
   - 4.1. Ausgewähltes Spitzen-Know-how
   - 4.2. Leitende Mitarbeiter mit über 15 Jahren Java-Erfahrung und eigenen anerkannten Publikationen
   - 4.3. Entwicklungsstandort Berlin - ausschließlich „Made in Germany“
   - 4.4. Integration von Projektmanagement in das Software-Entwicklungs-Team
   - 4.5. Konzeption IT-Lifecycle in der Applikationsplattform: Anforderung, Produktion, Test, Roll-Out, Abnahmen, Betrieb
   - 4.6. Integration von Ablaufspezialisten (Workflowsteuerung) in das Software-Entwicklungs-Team
5. ProSuite „Normen“ führen den Entwickler und halten diesen in dem vom Kunden gewünschten Korridor
6. Best Practice
   In Pilotprojekten wurde die Konzeption während der Entwicklung von ProSuite direkt getestet und erfolgreich im Markt real umgesetzt

6 EITCO und ProSuite

6.1 Das Team

Die EITCO bietet neben einer *State-of-the-Art* Entwicklung ein umfangreiches Team an erfahrenen IT-Beratern. Diese Berater haben größtenteils langjährige Erfahrung in der Analyse, Entwicklung, Implementierung und Roll Out. Weiterhin verfügen wir über umfangreiche Expertisen im Projekt-Anforderungs-, Test- und Qualitätsmanagement.

6.2 Die Effizienz

Vergleicht man die Effizienz zwischen der Entwicklung mit ProSuite bei gleichen Anforderungen und gleicher definierter geprüfter Qualität mit einer klassischen Anwendungsentwicklung, beträgt der **Effizienzquotient von ProSuite 1:5**. Durch die Vorfertigung von ProSuite müssen im Projekt nur noch die projektspezifischen Entwicklungen getätigt werden.

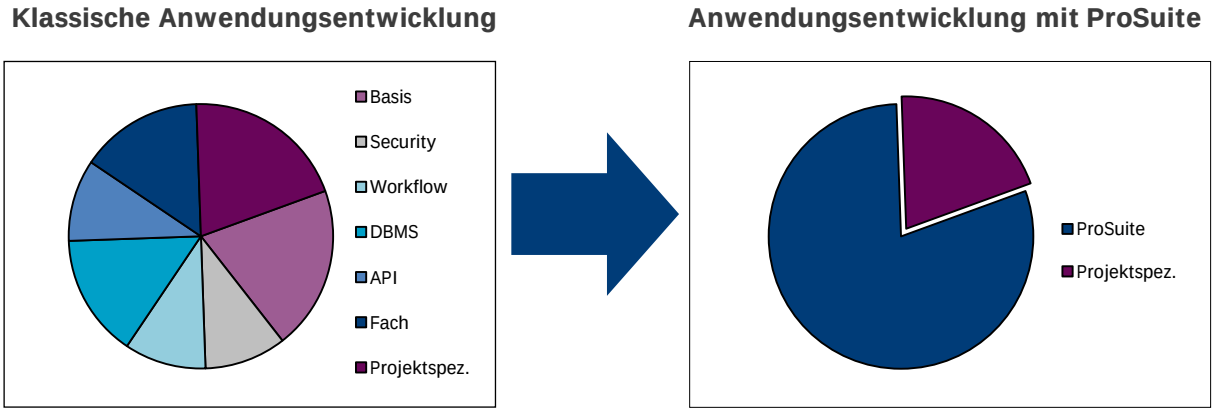

Abbildung 4: Effizienzquotient von ProSuite

An einem Beispiel verdeutlicht: Werden nach klassischer proprietärer Entwicklung 60 Entwicklermonate benötigt, so beträgt die Entwicklungszeit, abhängig von der Wahl der vorgefertigten Komponenten, mit ProSuite zehn Entwicklermonate.

Dies wurde in den ersten ProSuite Kundenprojekten bestätigt. Grund für diese Effizienz ist nicht alleine einfache Auswahl von Open Source Komponenten. Im Wesentlichen konnten wir diese Effizienz mit folgenden Eigenschaften erreichen:

1. Klare, qualitätsgeprüfte Architektur
2. Kluge Assemblierung
   2.1. Detaillierte Qualitätsprüfungen nach den ProSuite-Qualitätsstandards vor Verwendung von Komponenten

3. Multi-Channeling
4. Vorfertigung von Anwendungen
5. Kombination von Fach- und Technologiekompetenz
6. Integration von Projekt-Know-how in die ProSuite-Konzeption
7. Hoch qualifizierte Mitarbeiter, mit Sitz in Deutschland

6.3 Stärke im Projekt

EITCO ist es gewohnt, Kundenprojekte im größeren Umfang umzusetzen. Kundenbezogene Entwicklungsprojekte durchzuführen, in denen große Entwicklerteams zum Einsatz kommen, ist für uns geübte Praxis.

Weiterhin bieten sich beim Personaleinsatz in Projekten noch folgende Vorteile von ProSuite:

Einarbeitung
Durch die State-of-the-Art-Architektur basierend auf Standardtechnologien können Java-Entwickler effizient in ProSuite eingearbeitet werden.

Mitarbeiterperspektive
Java-Entwickler arbeiten gerne mit ProSuite, da diese modernen Konzepte die Mitarbeiter in der beruflichen Entwicklung voranbringen.

Kundenmitarbeiter
Durch die kurze Einarbeitungszeit in ProSuite können verfügbare Java-Entwickler eines Kunden auch unter wirtschaftlichen Aspekten kurzfristig in ProSuite eingearbeitet werden.

Ausbildung in der EITCO acadamy
Projektunabhängig können Java-Entwickler zertifiziert zum ProSuite-Entwickler ausgebildet werden.

7 Fazit

Mit ProSuite konnte die EITCO seit 2008 eine offene Java-Entwicklungsplattform zur individuellen Applikationsentwicklung aufbauen. Im Mittelpunkt standen nicht nur technologische Aspekte sondern vor allem **Zukunftssicherheit** und **Wirtschaftlichkeit**.

Die Zukunftssicherheit spiegelt sich in der Architektur wieder. In Kombination mit der fachlichen Kompetenz und den vorgefertigten Komponenten konnte bei der Entwicklung von ProSuite auch der gewünschte wirtschaftliche Effekt erreicht werden.

Eine Entscheidung für ProSuite und das ProSuite-Produktionsmodell ist eine Entscheidung für eine nicht-proprietäre Anwendungsentwicklung auf der Basis von Java und offen zugänglichem Know-how, womit eine hohe Herstellerunabhängigkeit erreicht wird.

Ziel von ProSuite ist es, Entscheidungen über Geschäftsmodelle, Kundenzugänge und Organisation bestmöglich zu unterstützen, indem Technologie und Fachlichkeit auf absolutem Spitzenniveau und hohem Grad an Vorfertigung zu effektiven und effizienten Anwendungen kombiniert werden. Die modulare Organisation von ProSuite-Anwendungen gibt Kalkulationssicherheit und sichert die langfristige Nutzung der Investitionen durch die Möglichkeit stetiger Evolution von Technologie und Business-Logik anstelle vorprogrammierter Veralterung und Ablösung.

ProSuite-Anwendungen vereinfachen die Integration in bestehende Landschaften durch innovative Schnittstellentechnologie, die ein nativer Bestandteil der Architektur ist.