

Fallstudiensammlung

Fallstudie Credit Suisse

Serviceorientierte Architektur

Roger Heutschi, Jan Schemm

Institut für Wirtschaftsinformatik,

Universität St. Gallen



Inhaltsverzeichnis

1 Credit Suisse 1
2 Ausgangssituation 1
2.1 Ursprüngliche IT-Architektur und Projektauslöser 1
2.2 Projektziele und organisatorische Verankerung 2
3 Implementierung einer serviceorientierten Architektur 3
3.1 Entwicklung zentraler Architekturrichtlinien und Organisationsstrukturen 3
3.1.1 Begriffsverständnis und Zielsetzungen 3
3.1.2 Organisationsstrukturen und Designprinzipien 5
3.1.3 Architekturcontrolling 5
3.2 Abgrenzung von Applikationsdomänen 6
3.3 Entwicklung der technischen Infrastruktur 8
3.3.1 Credit Suisse Information Bus 8
3.3.2 Credit Suisse Event Bus 9
3.3.3 Bulk Transfer Infrastructure 10
3.4 Projektgetriebene Serviceidentifikation und -implementierung 11
3.4.1 Serviceentwicklungs- und Reviewprozess 12
3.4.2 Servicespezifikation 15
4 Bisherige Erfahrungen und geplante Weiterentwicklungen 16
4.1 Aufwand und Nutzen der serviceorientierten Architektur 16
4.2 Herausforderungen und Erfolgsfaktoren 17
4.3 Geplante Weiterentwicklungen 18
5 Erkenntnisse aus der Fallstudie 19
Expertengespräche 21
Literatur 21

Zusammenfassung

Das Finanzdienstleistungsunternehmen Credit Suisse begann 1998 mit der Umsetzung einer Serviceorientierten Architektur (SOA). Das Unternehmen wollte damit Plattformen und fachliche Applikationsgruppen besser entkoppeln und eine schnelle Anpassung und Erweiterung der IT-Architektur bei gleichzeitig langfristiger Kosteneffizienz unterstützen. Die Architekturverantwortlichen wählten ein pragmatisches, evolutionäres Vorgehen und erweiterten die Integrations- und Servicearchitektur schrittweise. Nachdem die Umsetzung von Services bisher hauptsächlich im Rahmen fachlich getriebener Neuentwicklungen stattfand, werden sich zukünftige Architekturaktivitäten auf eine Komponentisierung der Mainframe-Applikationen und eine nachträgliche Anpassung bestehender Integrationsbeziehungen zwischen Applikationen konzentrieren.

© HSG / IWI / IWI2 / BN3 Cases

1 Credit Suisse

Die Credit Suisse Group ist ein führendes global tätiges Finanzdienstleistungs-Unternehmen mit Hauptsitz in Zürich. Sie bietet Privatkunden sowie kleineren und mittelgrossen Firmen umfassende Finanzberatung, Bankprodukte sowie Vorsorge- und Versicherungslösungen der Winterthur an. Im Bereich Investment Banking unterstützt sie globale Institutionen und Unternehmen, staatliche Körperschaften und Privatkunden als Finanzmarkt-Intermediär. Die Credit Suisse Group Namenaktien (CSGN) sind in der Schweiz sowie als American Depositary Shares (CSR) in New York kotiert. Die Gruppe beschäftigt weltweit rund 62'000 Mitarbeiterinnen und Mitarbeiter. Per 30. September 2005 verwaltete sie Vermögen in der Höhe von CHF 1'404.6 Mia.

CREDIT SUISSE GROUP	
Gründung	1856
Firmensitz	Zürich, Schweiz
Branche	Finanzen und Versicherungen
Firmenstruktur	Konzernstruktur mit den Geschäftseinheiten Credit Suisse, Credit Suisse First Boston sowie Winterthur
Geschäftsbereiche	Private Banking Privatkunden in der Schweiz Firmenkunden in der Schweiz Globales Investment Banking Winterthur: (Life & Pensions, Non-Life)
Reingewinn	5.628 Mrd. CHF (2004)
Mitarbeiter	60'500 (2004)
Homepage	http://www.credit-suisse.com

Tabelle 1-1: Kurzportrait Credit Suisse Group

2 Ausgangssituation

2.1 Ursprüngliche IT-Architektur und Projektauslöser

Wie bei anderen Grossunternehmen ist auch die IT-Architektur der Credit Suisse historisch gewachsen. Mit den in PL/I auf einem IBM Mainframe entwickelten terminalbasierten Systemen reichen die Wurzeln der heutigen Applikationslandschaft bis in die 70er Jahre zurück. Daneben existieren u.a. in Smalltalk entwickelte Client/Server-Applikationen aus den späten 80er Jahren und, mit dem Aufkommen des Inter- und Intranets in den 90ern, Systeme mit Multitier-Architekturen, z.B. auf Basis von Java. Wegen Zeit- und Kostendruck sowie fehlenden zentralen Designregeln und -prozessen entstanden in der Vergangenheit viele Ad-hoc-Lösungen. Bis Ende der 90er Jahre besass die IT-Architektur eine Komplexität, die

© HSG / IWI / IWI2 / BN3 Cases

Anpassungen nur mit unverhältnismässig hohem Aufwand erlaubte [vgl. Hagen 2004, 68]: Alleine für das Schweizer Bankgeschäft befanden sich rund 600 in verschiedenen Technologien entwickelte Applikationen im Einsatz, die unterschiedliche Lebenszyklen aufwiesen, Funktionen und Daten redundant implementierten und viele unkontrollierte Abhängigkeiten bzw. spezialisierte Punkt-zu-Punkt Schnittstellen untereinander besassen. Mehrere Projekte scheiterten in dieser Zeit mit hohen Verlusten. So wurde z.B. ein Projekt zur Entwicklung eines neuen Buchungssystems nach rund zwei Jahren ergebnislos gestoppt.

2.2 Projektziele und organisatorische Verankerung

Ende der 90er Jahre zeichnete sich eine umfangreichere Einführung Intranet- und Internet-basierter Applikationen ab. Die zentrale IT-Architektur unter der Verantwortung des CIO der Credit Suisse untersuchte verschiedene Optionen, um ein weiteres Komplexitätswachstum zu vermeiden, die IT-Systeme wieder beherrschbarer zu machen und die notwendigen Modernisierungen durchführen zu können (s. Tabelle 2-1).

Option	Diskussion
Kauf eines Standard-Softwarepakets	Da in der Schweiz nur eine weitere Bank mit ähnlichen Datenvolumen- bzw. Skalierbarkeitssanforderungen an eine Bankenplattform existierte, waren auf dem Markt keine passenden Angebote vorhanden. Internationalen Lösungen fehlte wesentliche Funktionalität.
Nutzung der Lösung einer anderen Bank	Wurde nur im Fall einer Fusion als realistisch angesehen und löste das Problem veralteter Technologien nicht, da die meisten Banken eine ähnliche Systemlandschaft besassen.
Komplette Neuentwicklung	Wurde aufgrund der Kosten (geschätzte 1 Mrd. CHF), der langen Entwicklungszeiten (5-7 Jahre) und wegen fehlenden Entwicklerkapazitäten verworfen.
Managed Evolution	Eine pragmatische, evolutionäre Anpassung und Erweiterung der Systemlandschaft (Anpassung bestehender Applikationen, Infrastrukturmodernisierung) mit strikten zentralen Steuerungsprozessen erlaubt die Nutzung der Stärken der bestehenden Systeme und des existierenden IT-Know-hows.

Tabelle 2-1: Untersuchte Architekturoptionen [vgl. Koch/Murer 1999, 194]

Credit Suisse entschied sich für einen Ansatz, den sie „Managed Evolution“ nennt. Dieser Ansatz sah vor, eine zentral gesteuerte Integrationsarchitektur nach den Prinzipien einer Serviceorientierten Architektur (SOA) schrittweise zu etablieren und Anpassungen bzw. Erweiterungen an bestehenden Applikationen einzelprojektgetrieben durchzuführen. Mit einer SOA sollten Strukturen geschaffen werden, die eine Entkopplung der verschiedenen Applikationen und fachlichen Applikationsgruppen und damit eine schnelle Anpassung und

© HSG / IWI / IWI2 / BN3 Cases

Erweiterung der Systemlandschaft erlauben und gleichzeitig die langfristige Kosteneffizienz sicherstellen.

3 Implementierung einer serviceorientierten Architektur

Credit Suisse begann 1998 mit der Umsetzung der Serviceorientierten Architektur. Die zentrale IT führte drei Teilprojekte zur Entwicklung der SOA durch:

- Die Entwicklung *zentraler Architekturrichtlinien und Organisationsstrukturen* definierte grundlegende Architekturziele sowie Standards, Methoden und Prozesse für das Design, die Verwaltung und die Qualitätssicherung von Services.
- Mit der Definition von *Applikationsdomänen* wurden Applikationen nach fachlichen Gesichtspunkten gruppiert. Die Domänenarchitektur bildet das wichtigste Hilfsmittel für Kopplungsentscheidungen zwischen Applikationen und die Frage, in welchen Fällen über Services integriert werden soll und wann nicht.
- Die Entwicklung einer *zentralen Integrationsinfrastruktur* schaffte die technische Basis für die domänenübergreifende, servicebasierte Applikationsintegration.

Übereinstimmend mit der „Managed-Evolution-Strategie“ fand keine architekturweite Planung der Service-Landschaft statt. *Services* wurden projektgetrieben und je nach Geschäftsanforderungen identifiziert und dann gebaut, wenn eine Client-Applikation eine bestimmte Funktionalität benötigte.

3.1 Entwicklung zentraler Architekturrichtlinien und Organisationsstrukturen

3.1.1 Begriffsverständnis und Zielsetzungen

Ausgangspunkt der Architekturentwicklung bildete ein gemeinsames Verständnis über die zentralen *Elemente und Begrifflichkeiten* einer SOA. Für Credit Suisse ist eine SOA Bestandteil der Integrationsarchitektur. Diese definiert Prinzipien, Prozesse und technische Lösungen für das Management der Verteilung und Heterogenität der Applikationslandschaft und der zugrunde liegenden technischen Plattformen [vgl. Hagen 2004, 72]. Die SOA bildet dabei eine Architekturschicht, die geschäftsgetriebene Neuentwicklungen und Entwicklungen zur

© HSG / IWI / IWI2 / BN3 Cases

Verbesserung der Architektureffizienz (z.B. System- und Technologieharmonisierungen) auf einer Abstraktions- und Gestaltungsebene zusammenführt. Wichtige Konzepte der Credit Suisse SOA sind Applikationsdomänen und Services. Eine Applikationsdomäne repräsentiert einen Teilbereich der Applikationslandschaft, der fachlich verwandte Applikationen zusammenfasst und öffentliche Schnittstellen auf die in ihr gekapselten Ressourcen bietet [vgl. Schlamann 2004, 6]. Diese Schnittstellen für Applikationen anderer Domänen werden mittels Services umgesetzt. Services stellen (im Vergleich zu Objekten) umfassendere grobgranulare Funktionen zur Verfügung und werden entweder direkt durch Servicenutzer oder bestimmte geschäftlich relevante Ereignisse in einer Applikation („Events“, z.B. Änderungen in der Datenbasis einer Stammdatenbank) ausgelöst. Wichtigster Aspekt eines Services ist für Credit Suisse die „gemanagte Schnittstelle“: Services sind einheitlich in einem öffentlichen Verzeichnis erfasst, dürfen nur in definierten Prozessen angepasst werden, müssen bestimmte Designprinzipien einhalten und qualitätsgeprüft sein. Im historischen Sprachgebrauch der Credit Suisse wird der Begriff „Service“ nur für die ursprünglichen synchronen, nach dem Request/Reply-Prinzip implementierten Domänenschnittstellen (s. Kap. 3.3) verwendet, und bezeichnet dort eine einzelne Operation. Prinzipiell lässt sich der Begriff aber unabhängig vom verwendeten Kommunikationsmechanismus auf sämtliche der „gemanagten“ Domänenschnittstellen (auch event- und filebasierte Schnittstellen) ausdehnen.

Für die neue Integrationsarchitektur wurden folgende *Ziele* bestimmt [vgl. Hagen 2004, 73]:

- Eine Reduktion der Komplexität von Beziehungen zwischen Applikationen, die sich in Kostenreduktionen bei Integrationsprojekten manifestiert,
- eine geeignete Kopplung zwischen Applikationen und Applikationsteilen, die weder zu vielen Abhängigkeiten noch zu Performanzproblemen für die Interaktion zur Laufzeit führt,
- grösstmögliche Wiederverwendung und geringe Redundanz von Funktionen und Daten und
- ein geeignetes Technologieportfolio, das so wenig Middleware-Produkte wie möglich umfasst aber gleichzeitig alle Projektanforderungen unterstützt.

© HSG / IWI / IWI2 / BN3 Cases

3.1.2 Organisationsstrukturen und Designprinzipien

Der evolutionäre, projektgetriebene Ansatz zur Entwicklung der Servicearchitektur birgt die Gefahr eines Service- und Technologiewildwuchses sowie von Redundanzen und Qualitätsmängeln in den entwickelten Lösungen. Um dies zu vermeiden, baute die zentrale IT *Organisationsbereiche* für die Entwicklung und den Betrieb der Integrationsinfrastruktur sowie für die Unterstützung und Qualitätssicherung von Fachprojekten auf und legte einen formalisierten, strikt durchgesetzten *Projektmanagementprozess* für fachliche Entwicklungsprojekte mit *Qualitätschecks* in verschiedenen Projektphasen fest (s. Kap. 3.4). Daneben definierte sie verschiedene technische und fachliche *Vorgaben für das Servicedesign*. Sie umfassen z.B. Vorgaben, in welchen Fällen Services zu entwickeln bzw. zu nutzen sind, wann die zentrale Integrationsinfrastruktur verwendet werden muss, zu verwendende Notationen und Kataloge für die Servicebeschreibung, Standards für Basis-Datentypen und eine Reihe von Prinzipien für das fachliche Servicedesign (s. Kap. 3.4). Diese Richtlinien werden aus den Erfahrungen in den Fachprojekten und dem laufenden Betrieb kontinuierlich erweitert. So zeigte sich z.B. mit steigendem Servicewachstum, dass Fachprojekte aufgrund einer unzureichenden Kategorisierung zunehmend Schwierigkeiten haben, geeignete bestehende Services zu identifizieren. Als Konsequenz werden gegenwärtig pro Applikationsdomäne die wichtigsten Informationsobjekte identifiziert, die zukünftig eine bessere fachliche Einordnung von Services erlauben sollen. Die Definition einheitlicher Fehlercodes auf Serviceebene für eine einfachere Servicenutzung stellt einen weiteren aktuellen Ausbau dar.

3.1.3 Architekturcontrolling

Zur laufenden *Überwachung und Steuerung der SOA* entwickelte die zentrale IT eine Architekturscorecard. Abgeleitet aus den Architekturzielen erhebt diese regelmässig u.a. folgende Messgrössen [vgl. Schwinn/Hagen 2006, 285ff.]:

- Resistance to change: Vergleich von Projektkosten im Zeitverlauf, mit einer Normierung der Projektkomplexität anhand von Use Case Points (vgl. dazu z.B. [Cohn 2005])
- Time to Market: Entwicklungszeiten pro Use Case Point
- Wiederverwendungsfaktor von Services
- Anzahl Services im Zeitverlauf

© HSG / IWI / IWI2 / BN3 Cases

- Anzahl Serviceaufrufe bzw. Messages pro Zeitintervall (Monat, Tag)
- Durch einzelne Domänen bearbeitete Serviceaufrufe pro Zeitintervall

3.2 Abgrenzung von Applikationsdomänen

Mit der Entwicklung einer Domänenarchitektur wollte die Credit Suisse die vielen hundert Applikationen verwaltbar machen und besser strukturieren. Eine Domäne fasst alle Applikationen und Daten zusammen, die zu einem bestimmten fachlichen Bereich gehören. Domänen bilden wichtige Anhaltspunkte für die zu wählenden Kopplungsgrade in Integrationsprojekten: Während innerhalb von Domänen enge Kopplungsbeziehungen zwischen Applikationen und Applikationsbestandteilen zulässig sind (z.B. durch Verwendung einer gemeinsamen Datenbank), wird zwischen Domänen eine konsequente Kapselung über wohldefinierte und stabile Services, die alle Details ihrer Implementierung verbergen, umgesetzt. Eine lose Kopplung über Services erlaubt unabhängige Lebens- bzw. Entwicklungszyklen zwischen Domänen und stellt für Credit Suisse einen Haupterfolgsfaktor für die Wiederverwendung und längerfristige Wartbarkeit der Applikationslandschaft dar.

Bei der Abgrenzung der Applikationsdomänen ging die zentrale IT von der bestehenden Applikationslandschaft aus. Sie erstellte einen Katalog mit den existierenden Applikationen und gruppierte sie nach ihrer fachlichen Zusammengehörigkeit. Die Interaktionshäufigkeit zwischen Applikationen diente dabei als wichtiger Anhaltspunkt. Die Applikationsarchitektur unterscheidet heute rund 20 Domänen. Sie sind primär nach Produkten und Kern-Bankenfunktionen, produktübergreifenden Funktionen und Daten, Supportfunktionen sowie externen Kommunikationskanälen strukturiert (s. Abbildung 3-1).

© HSG / IWI / IWI2 / BN3 Cases

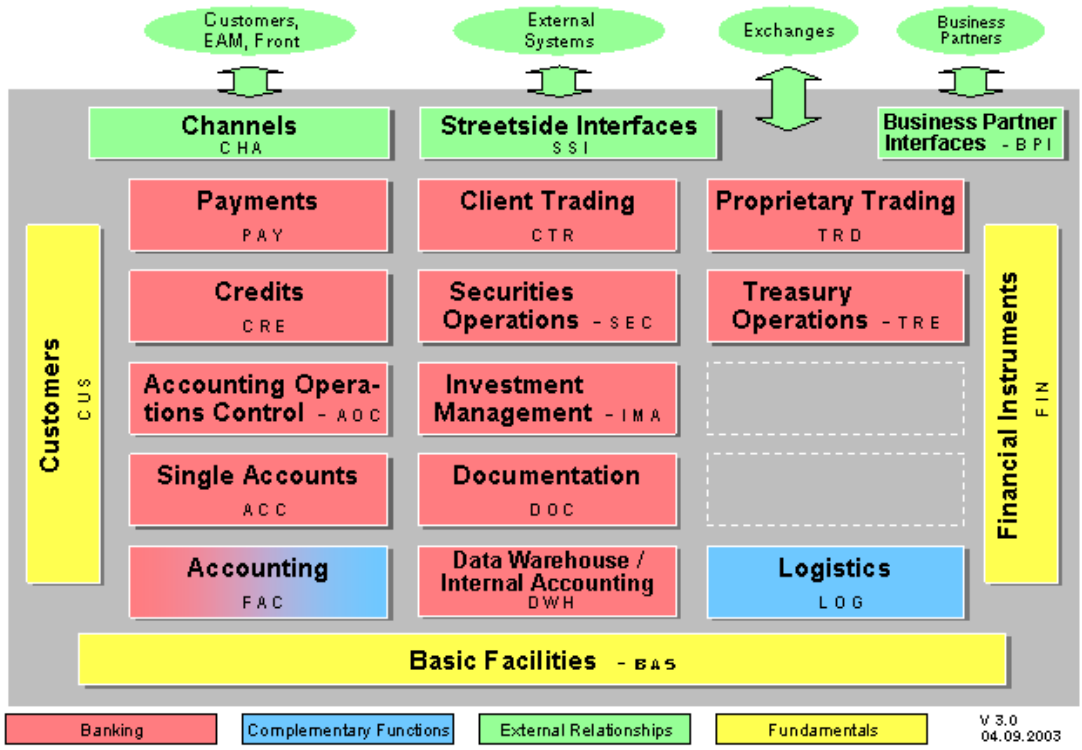


Abbildung 3-1: Domänenarchitektur der Credit Suisse (Quelle: Credit Suisse)

Die Domänenstruktur stellte ursprüngliche eine rein konzeptionelle Zielarchitektur für Neuentwicklungen dar. Aus Kosten- und Zeitgründen wurden Schnittstellen zwischen bestehenden Applikationen nicht nachträglich als Services umgesetzt bzw. nur dann, wenn sie im Rahmen einer Plattform- bzw. Technologieablösung sowieso neu implementiert werden mussten. Funktionen und Daten bestehender Applikationen wurden ohne neue fachliche Anforderungen im Rahmen der Domänenbildung auch nicht getrennt oder zusammengeführt. So wurden z.B. die auf dem Mainframe implementierten Applikationen bisher nicht entflochten, obwohl sie Funktionen und Daten, die fachlich in unterschiedliche Domänen gehören, eng koppeln. Eine Ausnahme bildet die Domäne „Data Warehouse“, die definiert wurde, um die Data Warehouse-Systeme der Credit Suisse zu konsolidieren. Sie wird nach Abschluss der Konsolidierung wieder in eine andere Domäne überführt.

Die Domänenarchitektur ist nicht nur ein Entscheidungshilfsmittel dafür, wann serviceorientiert zu integrieren ist, sondern stellt für Credit Suisse auch ein wichtiges organisatorisches Konzept dar: Jede Domäne mit ihren Services besitzt einen verantwortlichen Architekten, Architektur-Assessments werden pro Domäne durchgeführt und auch das IT-Budget wird auf Applikationsdomänen aufgeteilt.

© HSG / IWI / IWI2 / BN3 Cases

3.3 Entwicklung der technischen Infrastruktur

Die zentrale Integrationsinfrastruktur standardisiert die zulässigen Integrationsmechanismen und -technologien zwischen Applikationen unterschiedlicher Domänen und stellt die technische Basis für die Implementierung und Nutzung von Services dar. Credit Suisse hat die Integrationsinfrastruktur phasenweise entwickelt und ausgebaut, angefangen von einem Integrationsbus für synchrone Serviceaufrufe, über die Entwicklung eines asynchronen Event-Busses bis zur Ergänzung um einen Filetransfer-Bus für grosse Datenmengen. Parallel zur Entwicklung der Middlewareinfrastruktur standardisierte die zentrale IT auch Applikations- und Entwicklungsplattformen. Sie verfolgte damit u.a. das Ziel, die Middlewaredienste, die Werkzeugunterstützung und den fachlichen Support für die Nutzung der Integrationsinfrastruktur auf eine beschränkte Anzahl Technologien zu konzentrieren. Unterstützt sind heute eine Host-, eine Java- und eine Microsoft-Plattform sowie je eine Plattform für ERP- und Data Warehouse-Systeme.

3.3.1 Credit Suisse Information Bus

Anfang 1998 bis 1999 entwickelte die zentrale IT der Credit Suisse einen CORBA-basierten Service-Integrationsbus, den Credit Suisse Information Bus (CSIB). Der CSIB basiert im Kern auf Orbix von Iona Technologies und unterstützt synchrone Serviceaufrufe. Der Entwicklung ging eine Analyse voraus, die folgende Anforderungen ergab [vgl. Koch/Murer 1999]:

- Schnittstellendefinitionen müssen von Programmiersprachen und Plattformen unabhängig sein.
- Die Lösung sollte auf dem Markt verfügbar sein und auf einem verbreiteten Industriestandard basieren, der von Produkten unterschiedlicher Anbieter unterstützt wird.
- Die Lösung muss mehrere Programmiersprachen (PL/I, Java, Smalltalk etc.) unterstützen und alle relevanten Plattformen (u.a. OS/390, UNIX, NT) integrieren können.
- Sie muss einen Directory Service zur dynamischen Adressierung von Serviceaufrufen bieten, die Definition und Verteilung von Metadaten (z.B. Schnittstellendefinitionen) unterstützten und Systemmanagement-Dienste enthalten.

© HSG / IWI / IWI2 / BN3 Cases

Wichtige Gründe für den Entscheid einer Umsetzung mit CORBA waren u.a. die Plattform- und Programmiersprachenunabhängigkeit der Technologie und dass sich ein Hersteller finden liess, der eine CORBA-Implementierung für PL/I auf IBM-Mainframes erstellte.

3.3.2 Credit Suisse Event Bus

Der CSIB stellte bald eine leistungsfähige und breit genutzte Infrastruktur dar. Bereits 1999 enthielt er rund 35 Service, die von fünf Frontend-Applikationen genutzt wurden. Ende 2000 nutzten 21 Applikationen rund 170 Services. Bereits damals zeichnete sich jedoch ab, dass synchrone Funktionsaufrufe, wie sie der CSIB bot, nicht für alle Integrationsanforderungen ausreichten. Insbesondere bei der Integration zugekaufter Standardsoftware zeigten sich Grenzen. Diese Produkte setzten teilweise die Haltung eigener Daten voraus und besassen eigene interne Datenmodelle, was zu Datenredundanzen und dem Bedarf nach häufigen Datensynchronisationen und -transformationen führte. Der CSIB eignete sich schlecht für den Austausch solcher grösserer Datenmengen. Er bot weder Unterstützung loser Kopplungsmechanismen wie der asynchronen nachrichtenbasierten Kommunikation, noch Dienste für die garantierte einmalige Zustellung von Nachrichten, regelbasiertes Routing oder Nachrichtentransformationen. Credit Suisse implementierte deshalb bis 2002 den Credit Suisse Event Bus (CSEB), der einen Event-basierten, asynchronen Nachrichtenaustausch unterstützte. Events werden z.B. zu festgelegten Zeiten oder bei definierten Statusänderungen aufgrund eines Geschäftsereignisses (z.B. Registrierung eines neuen Kunden) ausgelöst. Die Kommunikation kann zwischen zwei Parteien (Peer-to-Peer) oder nach dem Publish/Subscribe-Muster (mehrere Nutzer „abonnieren“ einen Service) stattfinden und reduziert im Vergleich zum Request/Reply-Kommunikationsmuster den Netzverkehr, da die Aufrufe durch Servicenutzer entfallen.

Der CSEB basiert auf der IBM WebSphere Produktfamilie mit WebSphere MQ als nachrichtenbasierte Middleware und dem WebSphere Business Integration Message Broker für regelbasiertes Routing und Nachrichtentransformationen. Die zentrale IT legte beim Aufbau des CSEB grossen Wert darauf, bezüglich der Methoden und Prozesse zur Serviceentwicklung nahe am bestehenden CSIB zu bleiben. So gelten z.B. für das Design von Events wo sinnvoll die gleichen Grundsätze wie für asynchrone CORBA-Services. Auch die Projektmanagement- und Qualitätssicherungsprozesse sind für beide Servicearten dieselben. Ausserdem verwenden beide Infrastrukturen ein gemeinsames, auf einer SQL-Datenbank basierendes zentrales Service-Repository.

© HSG / IWI / IWI2 / BN3 Cases

3.3.3 Bulk Transfer Infrastructure

Gegenwärtig befindet sich als dritter Integrationsbus die Bulk Transfer Infrastructure (BTI) im Aufbau. Sie soll den im Bankgeschäft intern und zwischenbetrieblich verbreiteten filebasierten Massendatentransfer im Batch-Modus unterstützen. Da die technische Infrastruktur dafür bereits vorhanden ist, konzentrieren sich die Anstrengungen momentan auf die Umsetzung des Konzepts der „gemanagten Schnittstelle“ und der dazugehörigen Prozesse für diesen Schnittstellentyp. Ausserdem werden das zentrale Servicerepository erweitert sowie ursprünglich projektspezifisch und bilateral vereinbarte Dateistrukturen vereinheitlicht.

Abbildung 3-2 gibt einen Überblick über die umgesetzte Integrationsinfrastruktur. Während der CSIB hauptsächlich als Plattformüberbrückungstechnologie für die domänenübergreifende Integration zwischen Backend- und Frontendsystemen (z.B. zwischen einer mainframebasierten Depotverwaltung und einer in Java implementierten Online-Banking Applikation) zum Einsatz kommt, dienen der CSEB und die BTI eher der Integration zwischen Backendsystemen, die auf unterschiedlichen Plattformen laufen. Die Bus-Integrationsinfrastruktur wird durch eine Workflow-Infrastruktur ergänzt, welche die prozessorientierte Applikationsintegration unterstützt.

© HSG / IWI / IWI2 / BN3 Cases

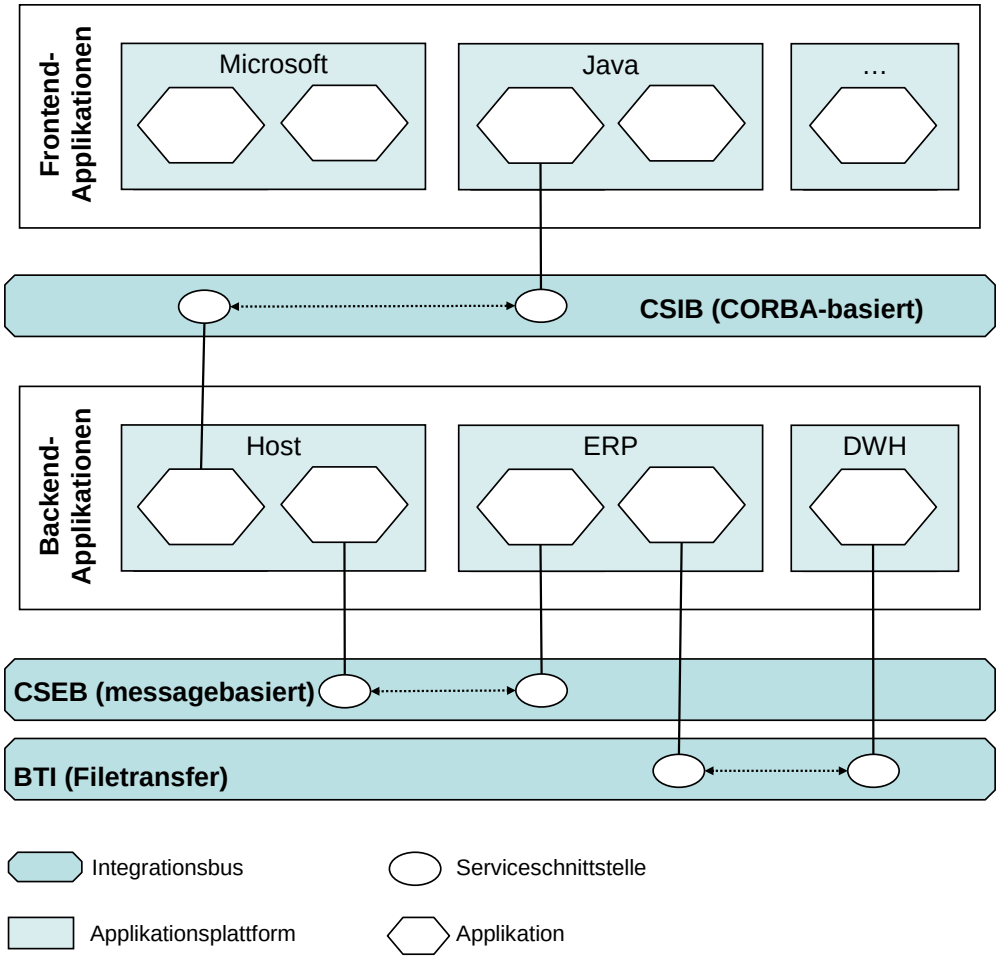


Abbildung 3-2: Credit Suisse Integrationsinfrastruktur

Momentan sind im CSIB rund 900 Services und im CSEB 20-30 Events implementiert. Beide weisen mit ca. 100 Mio. Serviceaufrufen bzw. Messages pro Monat (rund 5 Mio. täglich) eine vergleichbare Nutzung auf. Die zentrale IT verrechnet den Fachbereichen die Servicenutzung monatlich, allerdings nur anhand der Infrastrukturbelastung und nicht nach der Art bzw. Komplexität der einzelnen fachlichen Serviceleistung. Die Kosten für einen Servicenutzer errechnen sich beim CSIB nach der Anzahl Aufrufe, beim CSEB nach der ausgetauschten Datenmenge.

3.4 Projektgetriebene Serviceidentifikation und -implementierung

Bei der Serviceidentifikation und -entwicklung verfolgt Credit Suisse einen „bottom-up“-Ansatz: Services werden nicht zentral architekturweit geplant, sondern dann entwickelt, wenn ein Fachprojekt eine bestimmte Funktionalität benötigt. Da Credit Suisse 1999 verschiedene Intra- und Internet-Systeme entwickelte, konzentrierte sich die Serviceimplementierung in

© HSG / IWI / IWI2 / BN3 Cases

der Anfangsphase auf die Schnittstellen zwischen diesen Applikationen und den bestehenden Kernsystemen.

3.4.1 Serviceentwicklungs- und Reviewprozess

Um die organisatorisch verteilten Entwicklungsprojekte zu koordinieren und die Architekturkonformität der entstehenden Lösungen sicher zu stellen, definierte die zentrale IT einen standardisierten Projektmanagement- und Reviewprozess mit drei Qualitätschecks (s. Abbildung 3-3). Dabei kommen unabhängig vom verwendeten Bussystem bzw. Kommunikationsmechanismus die gleichen grundlegenden Prozesse, Methoden und Richtlinien zur Anwendung.

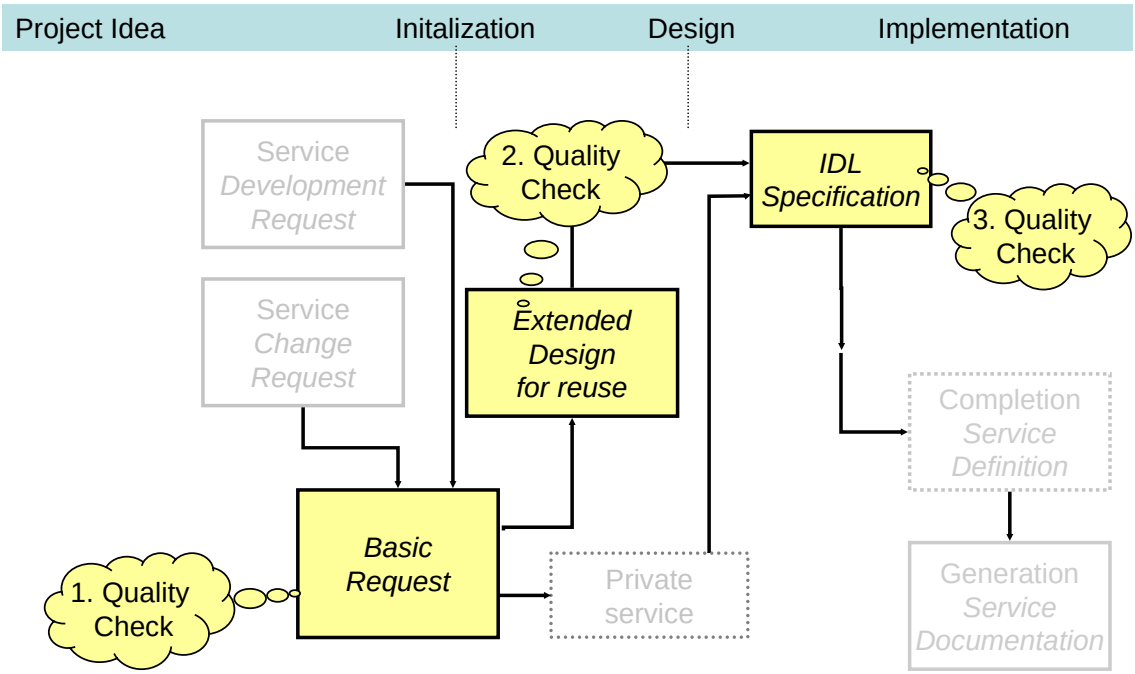


Abbildung 3-3: Projektreviewprozess (Quelle: Credit Suisse)

3.4.1.1 Qualitätscheck 1

Der Entwicklungs- und Reviewprozess beginnt damit, dass ein Fachprojekt im Rahmen einer Anforderungsanalyse aus Geschäftssicht die benötigten Funktionen der zu entwickelnden Lösung und ihre Effekte auf Daten beschreibt. Im Rahmen des Qualitätschecks 1 überprüft darauf das zentrale Architekturteam anhand des Servicerepository, ob bestehende Service wieder verwendet werden können, ob sie ggf. angepasst werden müssen oder ob die Funktionalität durch einen potenziellen neuen Service zur Verfügung gestellt werden muss. In die-

© HSG / IWI / IWI2 / BN3 Cases

sem Qualitätscheck entscheidet das Team, ob ein Service entwickelt oder angepasst wird und um welche Art von Service es sich handelt (Public oder Private, synchron, asynchron bzw. Event oder File). Grundlagen für diesen Entscheid stellen einerseits die *Domänenzugehörigkeit* der benötigten Funktionen oder Daten, die *technischen Plattformen* der potenziellen Serviceanbieter und -nutzer sowie *nichtfunktionale Eigenschaften* des zu unterstützenden Prozesses bzw. der Interaktion dar (s. Abbildung 3-4).

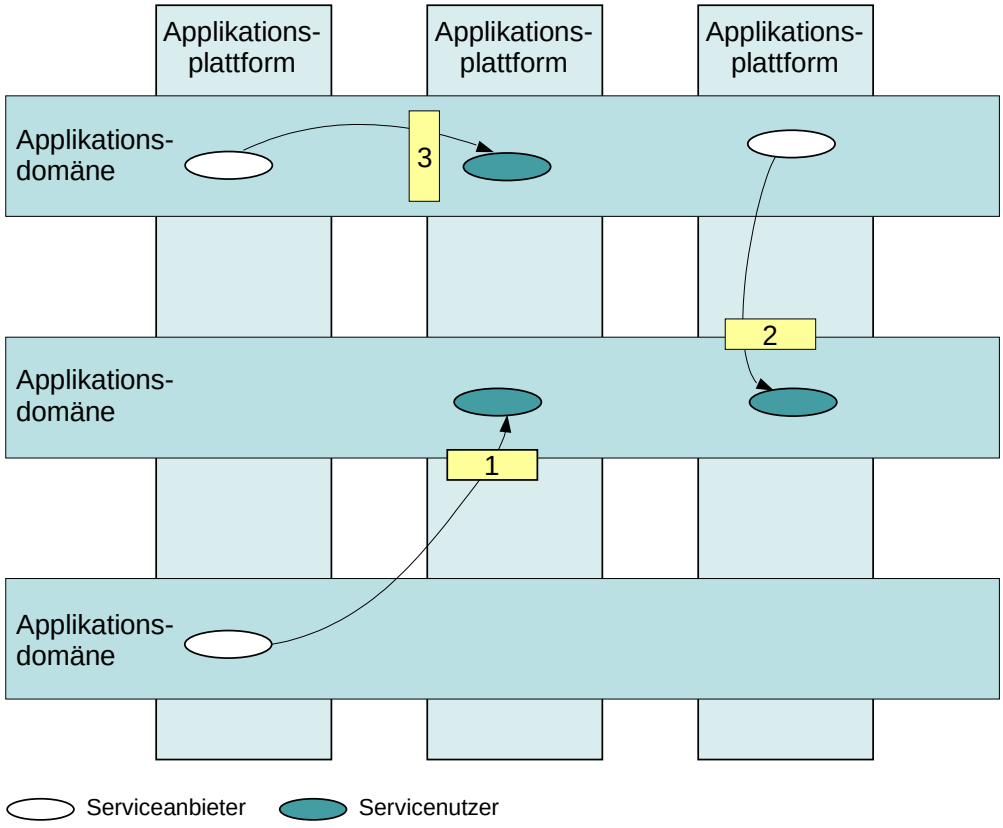


Abbildung 3-4: Entscheidungsgrundlagen für die Serviceentwicklung (Quelle: Credit Suisse)

„Public Services“ müssen zentral gemanagte Schnittstellen aufweisen, im zentralen Servicerepository registriert werden und durchlaufen den kompletten Reviewprozess mit allen Qualitätschecks. Sie werden dann entwickelt, wenn die Servicenutzer-Applikation Funktionen oder Daten benötigt, die sich in einer anderen fachlichen Domäne befinden (Schnittstellentypen 1 und 2 in Abbildung 3-4). Falls Servicenutzer und -anbieter dabei auf unterschiedlichen Applikationsplattformen implementiert sind (Typ 1), ist für die technische Integration zwingend die zentrale Integrationsinfrastruktur mit einem der drei Integrationsbusse (s. Kap. 3.3) zu verwenden. Falls sich Servicenutzer und -anbieter in unterschiedlichen fachlichen Domänen befinden, jedoch auf derselben Applikationsplattform implementiert sind (Typ 2), müssen sie

© HSG / IWI / IWI2 / BN3 Cases

nicht unbedingt über einen Bus kommunizieren sondern können plattformspezifische Mechanismen (z.B. JMS für Java-Applikationen) verwenden. Sie müssen aber die für Public Services vorgeschriebenen Designprinzipien einhalten und schnell an einen Bus angebunden werden können, falls sie von weiteren Nutzern auf anderen Applikationsplattformen wieder verwendet werden sollen. Befinden sich sowohl die servicenutzende als auch die serviceanbietende Applikation in derselben fachlichen Domäne, laufen aber auf unterschiedlichen Applikationsplattformen (Typ 3, z.B. Java-basiertes Benutzerfrontend für Datawarehousesysteme), spricht die Credit Suisse von einem „Private Service“. Ein Private Service wird nicht im zentralen Repository aufgeführt und verwaltet, unterliegt einem weniger strikten Reviewprozess und muss weniger Designvorgaben einhalten. Er wird jedoch über einen zentralen Integrationsbus zur Verfügung gestellt, weshalb der Qualitätscheck 3 auch für Private Services durchgeführt wird (s. Abbildung 3-3).

Im Rahmen des Qualitätschecks 1 entscheidet sich das Architekturteam auch dafür, ob eine Interaktionsbeziehung als synchroner CORBA-Service, als Event, als Fileschnittstelle oder mit einem gemischten Muster implementiert werden soll. Ein Beispiel für ein solches gemischtes Muster ist, dass ein Event bei einem Ereignis nur einen Schlüssel liefert und die Client-Applikation die benötigten Daten bei Bedarf über einen synchronen Serviceaufruf bezieht. Die zentrale IT ist momentan dabei, Heuristiken für die Wahl eines Kommunikationsmechanismus zu formalisieren. Wichtige Entscheidungsgrundlagen sind Zeitanforderungen (benötigt ein Servicenutzer die Daten sofort oder ist eine zeitlich verzögerte Verarbeitung möglich?) und das zu übermittelnde Datenvolumen.

3.4.1.2 Qualitätscheck 2

Den Qualitätscheck 2 führt das zentrale Architekturteam für Public Services während der Designphase des Entwicklungsprojekts durch. Er basiert auf einer detaillierteren Servicespezifikation, die eine Beschreibung der unterstützten Operationen und ihrer Effekte auf Daten, Input- und Outputdatenstrukturen, Ausnahmen sowie Vor- und Nachbedingungen der Servicenutzung enthält. Das Architekturteam prüft, ob das Servicedesign die vorgegebenen Prinzipien erfüllt. Gegenwärtig definiert die Architektur 20 Prinzipien für das Servicedesign, spezifische Prinzipien für Events und Fileschnittstellen befinden sich in Entwicklung. Tabelle 3-1 zeigt eine Auswahl der überprüften Designprinzipien.

© HSG / IWI / IWI2 / BN3 Cases

Prinzip	Erläuterung
Implementierungs-unabhängige Beschreibung	Serviceschnittstellen sollten während der Designphase technologie- und implementierungsunabhängig beschrieben werden. Dazu sind z.B. nur serviceintern verwendete, für einen Servicenutzer irrelevante Daten zu verbergen und standardisierte anstatt technologiespezifische Datentypen zu verwenden.
Beschreibung aus Geschäftsperspektive	Schnittstellen sind in fachlicher Terminologie zu beschreiben (z.B. nicht „Operation X liest Tabelle Y in Datenbank Z“).
Einhaltung von Namenskonventionen	Operations-, Informationsobjekt- und Parameternamen sollten bestimmte Namenskonventionen befolgen. Sie sind möglichst konkret zu benennen (z.B. „creditor“ anstatt „customer“, „purchaseDate“ anstatt „date“), Repetitionen in hierarchischen Datenstrukturen sind zu vermeiden und die Namen von boolschen Variablen sind so zu wählen, dass die Semantik von „wahr“ und „falsch“ eindeutig ist (z.B. „hasAccount“ anstatt nur „Account“).
Service-generalisierung	Services sollten so generalisiert werden, dass sie für einen möglichst breiten Nutzerkreis wieder verwendbar sind. Dazu sollte ein Service z.B. mehr Attribute liefern, als dies die spezifische Applikation für die er initial entwickelt wird, verlangt.
Servicegranularität	Um Netzwerkaufrufe zu minimieren und eine statusbehaftete Interaktion zwischen Service und Nutzer zu vermeiden, sollten Services in der Regel grobgranular sein, d.h. viel Funktionalität kapseln und möglichst alle benötigten Daten auf einmal liefern. Dabei sind je nach verwendetem Integrationsbus Beschränkungen bez. des ausgetauschten Datenvolumens zu beachten.
Redundanzfreie Input-/Output-Datenstrukturen	Abgesehen von Identifikationsschlüsseln sollte eine Wiederholung von Inputparametern in Output-Nachrichten vermieden werden.

Tabelle 3-1: Beispiele für Servicedesignprinzipien

Das Architekturteam protokolliert die Ergebnisse des Qualitätschecks und entscheidet über die Projektfortführung in die Implementierungsphase oder eine Überarbeitung des Designs.

3.4.1.3 Qualitätscheck 3

Qualitätscheck 3 wird in der Implementierungsphase für Public und Private Services durchgeführt. Middlewareexperten der zentralen IT überprüfen die korrekte Übersetzung der fachlichen Schnittstellenspezifikation in eine formale technische Schnittstellenbeschreibung (CORBA IDL im Falle von CORBA-Services, optional XML für Events) sowie die Performanzeigenschaften des Services unter Berücksichtigung der gewählten Integrationsinfrastruktur.

3.4.2 Servicespezifikation

Im Anschluss an die Entwicklung werden die Public Services im zentralen Repository erfasst. Die dort enthaltenen Servicespezifikationen sind nach einheitlichen Vorgaben strukturiert, werden im Laufe des Entwicklungsprozesses schrittweise erstellt und dienen jeweils als Input

© HSG / IWI / IWI2 / BN3 Cases

für die einzelnen Qualitätschecks. Eine vollständige Servicespezifikation enthält folgende Elemente:

- Eine formale, maschinenlesbare Schnittstellenspezifikation mit Operationen und Input- / Outputparametern (in IDL oder XML),
- eine knappe natürlichsprachliche Beschreibung der Servicefunktion, Operationen, Parameter und Effekte auf Daten,
- Ausnahmen und Vor- / Nachbedingungen (natürlichsprachlich, da eine formale Beschreibung in OCL zu kompliziert ist),
- angestrebte Servicelevels (Antwortzeitverhalten, Durchsatz, Reaktionszeiten im Fehlerfall),
- Planungsangaben (Status, Daten für durchgeführte oder geplante Tests, Datum der Produktivsetzung),
- Kontaktinformationen (Serviceentwickler, -betreiber und -nutzer),
- Implementierungsinformationen (Plattformen, Applikationen, Datenbanken).

4 Bisherige Erfahrungen und geplante Weiterentwicklungen

4.1 Aufwand und Nutzen der serviceorientierten Architektur

Credit Suisse benötigte für die *Entwicklung der einzelnen Integrationsbusse* jeweils 1.5 bis 2 Jahre, mit geschätzten Kosten von 1 bis 2 Mio. CHF. In der *zentralen IT* sind heute 5 Personen im Bereich Integrationsarchitektur beschäftigt, 2 bis 3 Personen betreiben und entwickeln die Integrationsinfrastruktur und 5 Personen unterstützen Fachprojekte bei der Serviceentwicklung und der Nutzung der Integrationsinfrastruktur. *Aufwände für Fachprojekte* entstehen durch die Entwicklung wieder verwendbarer Services und die zentralen Reviewprozesse. In der Regel verlängern sich Projekte in denen neue Services entwickelt oder bestehende erweitert werden um einige Personentage. Die anfallenden Kosten müssen von den Fachbereichen in den Projektbudgets einkalkuliert werden. Die zentrale IT geht allerdings davon aus, dass heute 80% bis 90% der von neuen Applikationen benötigten Services verfügbar sind.

© HSG / IWI / IWI2 / BN3 Cases

Nutzen der Serviceorientierten Architektur entsteht zum einen aus der *Wiederverwendung* von Services. Gegenwärtig werden die auf dem CSIB implementierten Services durchschnittlich 1.7-mal wieder verwendet, wobei einzelne Services (z.B. für das Lesen von Kundendaten) über 90 Nutzer aufweisen. Allerdings basiert diese Zahl auf einer freiwilligen Registrierung der Servicenutzer (die zentrale IT rechnet damit, dass effektiv rund 30% mehr Nutzer existieren) und bezieht sich auf die Wiederverwendung einzelner Operationen. Der Wiederverwendungsfaktor auf Ebene CORBA-Interface (umfasst im Schnitt rund 3 Operationen) beträgt 4.5. Credit Suisse rechnet damit, bis heute durch Wiederverwendung Entwicklungskosten von rund 30 Mio. CHF eingespart zu haben.

Weiter entstanden Einsparungen im *IT-Betrieb* durch die Harmonisierung und Standardisierung von Integrations- und Entwicklungsplattformen sowie einer besseren Entkopplung der fachlichen Applikationsdomänen. Insgesamt sind die IT-Aufwände in den letzten 5 Jahren bei wachsender Produktivität um rund 30% gesunken. Der Beitrag der SOA bzw. Integrationsarchitektur an dieser Kostenreduktion lässt sich nicht direkt bestimmen, dürfte aber signifikant sein.

Daneben sieht die Credit Suisse verschiedene *qualitative*, schlecht quantifizierbare Vorteile. Dazu zählen etwa die verbesserte Kommunikation zwischen zentraler IT und Fachbereichen oder höhere Flexibilität und kürzere Time-to-Market für neue Lösungen. Letztere wird für IT-Projekte seit einem Jahr erhoben (s. Kap. 3.1.3), Nutzeneffekt der SOA lassen sich bisher nicht genau beziffern. Die zentrale IT geht jedoch von Verkürzungen der Projektlaufzeiten von bis zu einigen Wochen oder Monaten aus. Dies ist u.a. darauf zurückzuführen, dass sich bei einer Wiederverwendung existierender Services Projektverzögerungen durch personelle Ressourcenengpässe vermeiden lassen.

4.2 Herausforderungen und Erfolgsfaktoren

Die Herausforderungen bei der Umsetzung der SOA waren hauptsächlich technischer und organisatorischer Natur. Die Servicearchitektur nahm in so kurzer Zeit eine so wichtige Rolle in Entwicklungsprojekten ein, dass es zu Performanzproblemen kam, die sich sofort negativ auf die Akzeptanz bei den Fachprojekten auswirkte. Neben solchen technisch bedingten Akzeptanzproblemen hatte die Architektur zu Beginn auch Mühe, die Fachbereiche vom Nutzen und der Notwendigkeit der neuen Architekturrichtlinien und Entwicklungsprozesse zu überzeugen. Die starke Unterstützung durch das Top-Management war wesentlich für ihre Durch-

© HSG / IWI / IWI2 / BN3 Cases

setzung. Eine weitere Herausforderung stellte die Auswahl der Architekturrichtlinien und -standards dar. Die im Laufe der Serviceentwicklung und -nutzung gesammelten Erfahrungen zeigten, dass zu Beginn z.B. die Standardisierung von Fehlercodes oder fachlichen Datenmodellen auf Serviceebene vernachlässigt wurde. Entsprechende Standards werden gegenwärtig entwickelt und nachträglich eingeführt (s. Kap. 3.1.2).

Wichtige Erfolgsfaktoren für die Umsetzung einer SOA sind für Credit Suisse:

- Die Konzentration auf die Definition sauberer, fachlich definierter und qualitätsgeprüfter Schnittstellen,
- eine strikt durchgesetzte Projektkontrolle,
- die Entscheidungsautorität der zentralen IT bei der Serviceentwicklung,
- starke Unterstützung durch das Top-Management,
- die personelle und werkzeugtechnisch in Entwicklungsplattformen integrierte Unterstützung der Fachprojekte (z.B. durch Codegeneratoren für die Serviceentwicklung),
- eine solide technische Integrationsinfrastruktur und ihre laufende Weiterentwicklung sowie
- ein kontinuierliches Architekturcontrolling über Metriken und Scorecards, um Entwicklungsbedarf zu erkennen und ggf. steuernd einzugreifen.

4.3 Geplante Weiterentwicklungen

Die Credit Suisse besitzt mit den zentralen Architekturrichtlinien und Organisationsstrukturen, der implementierten Integrationsinfrastruktur und den entwickelten Services eine leistungsfähige SOA. Momentan besteht jedoch noch Entwicklungsbedarf bei der nachträglichen Entkopplung der Applikationsarchitektur. Wegen des pragmatischen, projektgetriebenen Vorgehens bei der Serviceentwicklung existieren auch heute noch Redundanzen und unkontrollierte enge Kopplungsbeziehungen zwischen älteren Applikationen unterschiedlicher Domänen. Insbesondere die Applikationen auf der Host-Plattform sind teilweise noch eng miteinander verflochten. Um in Zukunft einzelne Elemente der Applikationslandschaft möglichst unabhängig entwickeln und mit geringem Aufwand austauschen zu können, will Credit Suisse Applikationsdomänen und Applikationen nachträglich stärker in Komponenten zerlegen

© HSG / IWI / IWI2 / BN3 Cases

und konsequent kapseln sowie alle Schnittstellen und Abhängigkeiten von anderen Komponenten und das nach aussen sichtbare Verhalten genau spezifizieren. Herausforderungen entstehen dabei durch ein jahrzehntelang unkontrolliertes Wachstum der Applikationslandschaft und teilweise verlorenes Wissen über die Semantik von Applikationen und ihrer Querbeziehungen.

Daneben werden Architekturrichtlinien und Servicedesignprinzipien sowie die Integrationsarchitektur laufend weiterentwickelt und neuen technischen Möglichkeiten angepasst. CORBA ist als Technologie immer weniger verbreitet. Dadurch steigen für Anwender die Kosten für den Betrieb und das Risiko einer Verdrängung der Hersteller vom Markt. Credit Suisse führt deshalb laufend Potenzialanalysen für einen Technologiewechsel auf Web Services durch. Gegenwärtig können Web Services aber noch nicht alle Anforderungen an die Integrationsinfrastruktur erfüllen. Lücken bestehen z.B. in der Performanz und Skalierbarkeit von Web Service-Lösungen sowie bezüglich Sicherheitsanforderungen. So fehlen z.B. geeignete Mechanismen für die Adressauflösung und das dynamische Routing von Serviceaufrufen und Mechanismen für das Durchreichen der Identität eines Servicenutzers in die Backendsysteme sind noch nicht ausreichend standardisiert bzw. in den Produkten der Softwarehersteller implementiert.

5 Erkenntnisse aus der Fallstudie

Das Fallbeispiel Credit Suisse zeigt einen erfolgreichen Ansatz für eine pragmatische und evolutionäre Einführung einer SOA. Dies betrifft sowohl den schrittweisen Auf- und Ausbau der zentralen Integrationsinfrastruktur und der Architekturrichtlinien als auch die geschäftsbedarfsgetriebene, dezentrale Serviceidentifikation und -entwicklung in einzelnen Fachprojekten. Dabei kommt einem koordinierenden zentralen Architekturmanagement und einer technischen und fachlichen Architekturstandardisierung grosse Bedeutung zu.

Das projektgetriebene Vorgehen zur Serviceidentifikation und -entwicklung hilft bei der schnellen und an den Prioritäten des Geschäfts orientierten Verbreitung der SOA im Unternehmen. Für die Realisierung verschiedener Nutzenpotenziale, wie z.B. die Verringerung von IT-Kosten durch Redundanzreduktion oder die Fähigkeit, Elemente der Applikationslandschaft schnell und mit geringen Auswirkungen auf andere Applikationen herauszulösen, sind aber auch Aufräumarbeiten in der bestehenden Applikationsarchitektur

© HSG / IWI / IWI2 / BN3 Cases

notwendig. Applikationen müssen u.U. mit grossem Aufwand zerlegt oder zusammengeführt und bestehende Beziehungen und Schnittstellen ersetzt werden.

Die Fallstudie verdeutlicht, dass sich die Einführung einer SOA insbesondere während der Aufbauphase, in der viele Service erst entwickelt werden müssen, nicht aus Sicht eines einzelnen Fachprojekts sondern nur aus Sicht der Gesamtarchitektur und längerfristig rechtfertigen lässt. Wegen der hohen Vorausinvestitionen in die Entwicklung organisatorischer Strukturen, Architekturrichtlinien und einer technischen Integrationsinfrastruktur lassen sich kurzfristige Gewinne kaum realisieren. Credit Suisse rechnet mit einer Dauer von rund vier Jahren, bis eine entsprechende zentrale Infrastruktur etabliert ist und sich rechnet.

© HSG / IWI / IWI2 / BN3 Cases

Expertengespräche

Claus Hagen, Zentrale Integrationsarchitektur, Credit Suisse AG, Zürich, 13.10.2005

Literatur

[Cohn 2005]
Cohn, M., Estimating With Use Case Points, in: Methods & Tools, 13, 2005, Nr. 3, S. 3-13

[Hagen 2004]
Hagen, C., Integrationsarchitektur der Credit Suisse, in: Aier, S., Schönherr, M. (Hrsg.), Enterprise Application Integration - Flexibilisierung komplexer Unternehmensarchitekturen, GITO Verlag, Berlin 2004, S. 61-83

[Koch/Murer 1999]
Koch, T., Murer, S., Service architecture integrates mainframes in a CORBA environment, Proceedings of the third International Enterprise Distributed Object Computing Conference, EDOC '99., IEEE, 1999, S. 194-203

[Schlamann 2004]
Schlamann, H., Service Orientation: An Evolutionary Approach, in: Cutter IT Journal, 17, 2004, Nr. 5, S. 5-13

[Schwinn/Hagen 2006]
Schwinn, A., Hagen, C., Measured Integration – Metriken für die Integrationsarchitektur, in: Schelp, J., Winter, R. (Hrsg.), Integrationsmanagement, Springer, Berlin et al. 2006, S. 267-292

© HSG / IWI / IWI2 / BN3 Cases