Technische Hochschule Ingolstadt

Probeklausur Integrations- und Migrationstechnologien

2019/2020

1. Motivation, Heterogenität, Integrationsart

1a) Beschreiben sie das Konzept einer horizontal organisierten IT. Erläutern Sie die Vor- und Nachteile. Wann ist diese Architekturform sinnvoll?

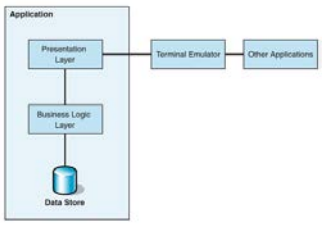
- Keine Abkehr von den Basissystemen
- Aber: Trennung/Entflechtung der Systeme von Prozess- und Oberflächenlogik
- Definition und bereitstellen von grobgranularen Fachdiensten
- Beliebige Verwendung der Dienste in unterschiedlichen Prozessen
- Eine einheitliche GUI welche den Benutzer durchgängig unterstützt
- Flexibilität und anpassarkeit

1b) Beschreiben Sie die sog. „syntaktische Heterogenität". Geben Sie zwei mögliche Beispiele an und beschreiben Sie, wie diese die Integration erschweren können.

- Technische Heterogenität
- Unterschiede auf der Ebene der technischen Infrastrukturen wie DBMS, Hardwareplattformen, Betriebssysteme und Netzwerkkomponenten

- Technische Heterogenität
  - Unterschiedliche Darstellung desselben Sachverhalts
  - Dezimalpunkt oder –komma
  - Euro oder €
  - Comma-separated oder tab-separated

1c) Erläutern Sie die Grundidee der „Presentationintegration". Geben Sie je einen Vor- und einen Nachteil an. Wann ist diese Integrationsform gut geeignet?



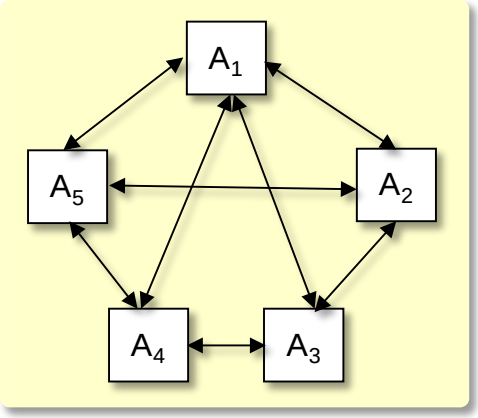
- When integrating multiple applications, you should minimize changes to existing systems because any change to an existing production system represents a risk and might require extensive regression testing.
- Many applications feature little or no separation between business and presentation logic. The only remote components of these applications are simple display and input devices. These display terminals access the central mainframe computer, and the central mainframe computer hosts all business logic and data storage.
- The business logic layer of many applications is programming-language specific and is not available remotely, unless it was developed on top of a specific remoting technology such as DCOM or Common Object Request Broker Architecture (CORBA).
- Directly accessing an application's database layers can cause corruption of application data or functionality.
- Integration von Anwendungen durch Simulation eines Benutzers
- Ein Bildschirmemulator liefert Eingabe an die zu steuernde Oberfläche
- Die Ausgabe des integrierten Programms wird abgefangen und strukturiert bereitgestellt.
- Einsatzbereich: Integration von Mainframes und Legacysystemen, Webanwendung
- Zugriff wird über eine Standard API bereitgestellt

2. Grundlagen

2a) Beschreiben Sie zwei Qualitätsmerkmale, die geeignet sind um eine Integrationsarchitektur zu beurteilen. (Fokus Unternehmensarchitektur)

- Erweiterbarkeit: Neue Systeme sollen mit wenig aufwand in der Landschaft betrieben warden können
- Robustheit: Wie Robust ist die Landschaft gegen Ausfall einzelner Systeme
- Austausch
- Komplexität

2b) Beschreiben Sie die Integrationsarchitektur „Point-To-Point“. Erstellen Sie hierzu eine Skizze. Gehen Sie auf Vorteile und Nachteile dieser Integrationsarchitektur ein. Wann stößt diese Architektur an Grenzen?



• Verbund unabhängiger Systeme, die durch ein Netzwerk miteinander verbunden sind.
• Alle Systeme sind gleichberechtigt und können sowohl Dienste konsumieren als auch bereitstellen.
• Keine zentrale Datenhaltung, jedes System hat seinen eigenen Datenbestand
• Komplexität: n * (n – 1)/2

Stärken:

- Geringe Start-/Infrastrukturkosten
- Autonome Systeme

Schwächen:

- Nur bei weinigen Systemen und wenigen Verbindungen praktikabel
- Einzelne Systeme nur mit hohem Aufwand austauschbar
- Sehr unflexibel
- Wiederverwendbarkeit von Komponenten ist beschränkt.
- Aufwändiger Betrieb

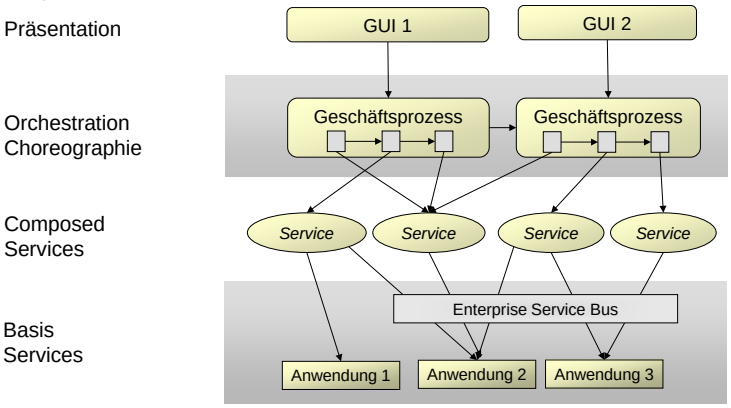
Chancen:

- Funktionserweiterungen innerhalb der Systeme können Schnell an neue Anforderungen angepasst werden

Gefahren:

- Hohe Folgekosten
- Fehlende Standardisierung

3) Fortgeschrittene Integrationsarchitektur



3a) Erläutern Sie das Grundprinzip einer Service-Orientierten-Architektur anhand der obigen Grafik. Erläutern Sie die wesentlichen Bestandteile.

Basis Services: Basisdienste einer Service orientierten Architektur. Stammdatendienst, Transformation

Composed Services/Intermediate Service: Kombinieren verschiedene Services zu einer neuen Gesamtfunktionalität. Kurzlaufende Aktivitäten die mehere Services umfassen.

Anwendungsservices: Geschäftsservice die komplett oder teilweise durch IT ausgeführt werden können, Mensch & Maschine ist bei der Ausführung beteiligt

3b) Begründen Sie, warum diese Architekturform zu einer höheren Flexibilität im Bereich der Orchestration und Composed Services führt?

Flexibilität: Austausch von Basisdiensten bei gleicher Schnittstelle jederzeit möglich. Definition von neuen Anwendungsservices flexibel möglich

Flexibilität: Proesse können einfach durch anpassung der orchestration geändert werden

3c) Erläutern Sie, warum das SOA-Prinzip „Benutzung von Standards“ für die Umsetzung einer SOA notwendig ist.

Vereinfachung der Integrationsaufgabe
Reduzieren der Heterogienität

4. Fehlerbehandlung in verteilten Systemen

4a) Beschreiben Sie das Konzept „Idempotenz". Wie kann dieses Konzept helfen, die Stabilität eines verteilten Systems positiv zu beeinflussen. Wie kann dieses Konzept in einer Applikation realisiert werden?

- Als idempotent bezeichnet man Funktionsaufrufe, die immer zu den gleichen Ergebnissen führen, unabhängig davon, wie oft sie mit den gleichen Daten wiederholt werden. Idempotente Arbeitsgänge können zufällig oder absichtlich wiederholt werden, ohne dass sie nachteilige Auswirkungen auf den Computer haben.

+ Umsetzung in applikation

4b) Nachfolgend soll das Zusammenspiel verschiedener Client- und Server-Strategien im Falle eines Serverabsturzes bewertet werden. Der Client sendet eine Nachricht an den Server. Dieser verarbeitet die Nachricht (Versand einer Mail) und antwortet dem Client mit einer Durchführungsnachricht. Abhängig von der Fehlerstrategie des Clients und der Ausführungsstrategie des Servers können unterschiedliche Ergebnisse beobachtet werden. (OK = Mail wird einmal versandt, DUP = Mail wird zweimal versandt, NULL = Mail wird nicht versandt)
Randbedingungen Client:
- Der Client wird über einen Absturz informiert, hat aber keine Information über den Status der Operation
- Strategien im Fehlerfall:
  - niemals eine neue Anforderung absetzen
  - immer eine neue Anforderung absetzen
  - neue Anforderung wenn keine Bestätigung der Auslieferung
  - neue Anforderung wenn eine Bestätigung der Auslieferung

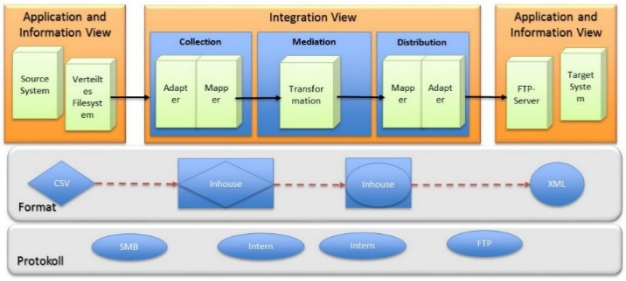
Randbedingungen Server:
- Durchführungsnachricht (M), Email versenden (P), Absturz/Crash (C)
- Klammern: Ereignis passiert nicht, da bereits Absturz

Tragen Sie das Ergebnis OK, DUP, NULL für jede Kombination in die nachfolgende Tabelle ein.

Client	Server					
	Strategie M→P			Strategie P→M		
Strategie zum erneuten Absetzen	MPC	MC(P)	C(MP)	PMC	PC(M)	C(PM)
Immer	DUP	OK	OK	DUP	DUP	OK
Nie	OK	NULL	NULL	OK	OK	NULL
Nur nach Bestätigung	DUP	OK	NULL	DUP	OK	NULL
Nur, falls keine Bestätigung erfolgt ist	OK	NULL	OK	OK	DUP	OK

5. Integration Architecture

Analysieren Sie das nachfolgende Beispiel einer möglichen Stammdatenverteilung.



5a) Erläutern Sie die Konzepte „Inhouse Format“ und „Externes Format“. Stellen Sie die Unterschiede zwischen den verschiedenen Formaten klar heraus. Markieren Sie die Formate in der obigen Skizze. Geben Sie ein sinnvolles Beispiel an, welches den Unterschied/Zusammenhang erläutert.

Inhouse-Format:

- Internes Format der Integrationslösung welches benutzt wird um die externen Nachrichten intern zu präsentieren.

Externes Format:

- Format welches von externen Protokollen genutzt wird um die Daten zu übertragen

Das externe Format wird durch einen Mapper in das interne Format überführt.

5b) Erläutern Sie den Unterschied zwischen der Komponente Transformation und der Komponente Mapper. Stellen Sie den Bezug zu den Inhouse- und Externen Formaten her.

Mapping:
Abbildung externer Formate auf das Interne Format

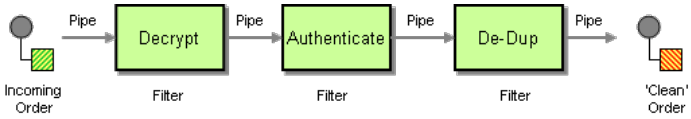
Transformation:
Umwandlung eines internen Formats in ein anderes Format

5c) Welche Aufgabe wird durch das sog. Routing erledigt?

Zustellen von Nachrichten zu definierten Empfängern/Komponenten innerhalb der Integrationslösung

6. Enterprise Integration Patterns

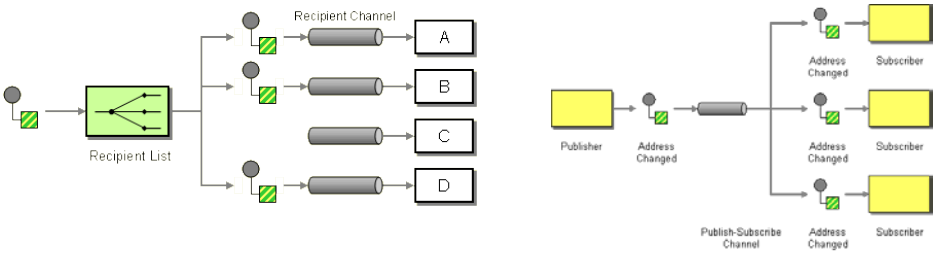
6b) Erläutern Sie das nachfolgend dargestellte „Pipe-And-Filter"-Muster. Nennen Sie mindestens zwei Vorteile (+Begründung), welche durch den Einsatz dieses Musters erreicht werden.



Use the *Pipes and Filters* architectural style to divide a larger processing task into a sequence of smaller, independent processing steps (Filters) that are connected by channels (Pipes).
Each filter exposes a very simple interface: it receives messages on the inbound pipe, processes the message, and publishes the results to the outbound pipe. The pipe connects one filter to the next, sending output messages from one filter to the next. Because all component use the same external interface they can be *composed* into different solutions by connecting the components to different pipes. We can add new filters, omit existing ones or rearrange them into a new sequence -- all without having to change the filters themselves. The connection between filter and pipe is sometimes called *port*. In the basic form, each filter component has one input port and one output port.

Skalierbarkeit + flexisbiliät

6b) Erläutern die Funktionsweise der beiden Muster „Recipient-List" und „Publish-Subscribe-Channel". Welcher Unterschied besteht zwischen den Mustern? Geben Sie für jedes Pattern ein sinnvolles Einsatzbeispiel an.

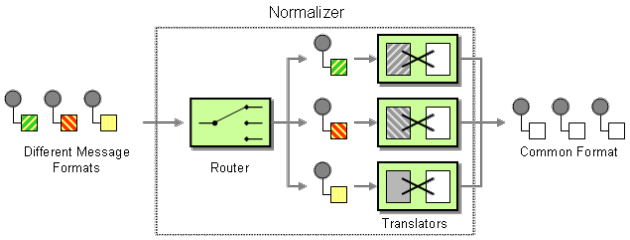


Zustellen einer Nachricht an einen oder merher empfänger

Recipient List:
Der Sender entscheidet wer die Nachrichten bekommnt

Publish-Subscribe:
Nachricht wird an alle registrierten Empfänger übermittelt

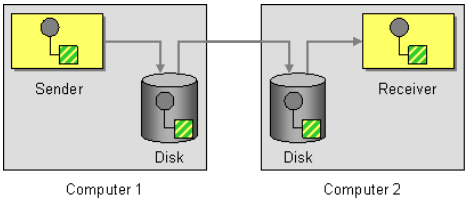
6c) Erläutern Sie das Muster „Normalizer". Geben Sie ein mögliches Beispiel an, welches den sinnvollen Einsatz verdeutlicht. Welches Architekturprinzip wird dadurch realisiert? Welcher Vorteil ergibt sich dadurch? Wann sollte das Muster eingesetzt werden?



The Normalizer uses a Message Router to route the incoming message to the correct Message Translator. This requires the Message Router to detect the type of the incoming message. Many messaging systems equip each message with a type specifier field in the Message Header to make this type of task simple. However, in many B2B scenarios messages do not arrive as messages compliant with the enterprise's internal messaging system, but in diverse formats such as comma separated files or XML document without associated schema. While it is certainly best practice to equip any incoming data format with a type specifier we know all to well that the world is far from perfect. As a result, we need to think of more general ways to identify the format of the incoming message. One common way for schema-less XML documents is to use the name of the root element to assume the correct type. If multiple data formats use the same root element, you can use XPATH expressions to determine the existence of specific sub-nodes. Comma-separated files can require a little more creativity. Sometimes you can determine the type based on the number of fields and the type of the data (e.g. numeric vs. string). If the data arrives as files, the easiest way may be to use the file name or the file folder structure as a surrogate Datatype Channel. Each business partner can name the file with a unique naming convention. The Message Router can then use the file name to route the message to the appropriate Message Translator.

- Kanonosches Datenmodell

6d) Erläutern Sie das Muster „Guaranteed-Delivery". Geben Sie ein mögliches Beispiel an, welches den sinnvollen Einsatz verdeutlicht. Welche Auswirkung hat der Einsatz dieses Musters auf die Integrations-Lösung? Wann sollte es eingesetzt werden wann nicht?



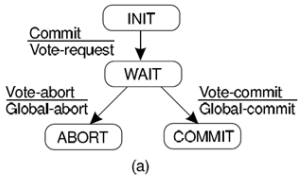
Use *Guaranteed Delivery* to make messages persistent so that they are not lost even if the messaging system crashes.

With *Guaranteed Delivery*, the messaging system uses a built-in data store to persist messages. Each computer the messaging system is installed on has its own data store so that the messages can be stored locally. When the sender sends a message, the send operation does not complete successfully until the message is safely stored in the sender's data store. Subsequently, the message is not deleted from one data store until it is successfully forwarded to and stored in the next data store. In this way, once the sender successfully sends the message, it is always stored on disk on at least one computer until is successfully delivered to and acknowledged by the receiver.

7 Transaktionen

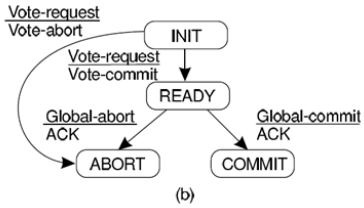
7a) Beschreiben Sie den Ablauf des 2-Phase-Commit-Protocoll's. Erläutern Sie hierzu die Zustandsautomaten der beteiligten Komponenten:

Koordinator



- Commit-Aufforderung wird an koordinator gesendet
- Vote-request an die teilnehmer senden
- Wenn alle Teilnehmer für commit stimmen dann global-commit an die Teilnehmer senden
- Andernfalls wird ein global-abort gesendet

Teilnehmer



- Bei einem Vote-Request über die transaktion abstimmen (entweider vote-commit oder vot-abort)
- Bei vote-commit übergang nach ready und auf globale Entscheidung warten: Wichtig: Zustannd muss immer „commited" werden können. Änderung der Meinung nicht möglich
- Abhängig von Koordinator wird die TA erfolgreich oder fehlerhaft beendet

7b) Wann kann die Transaktionsart „2-Phase-Commit" sinnvoll eingesetzt werden und wann nicht? In welchem Szenario führt diese Transaktionsart eher zu Problemen? Welche Alternative existiert? Geben Sie für beide Varianten die Art der Kopplung (Hoch/Niedrig) an.

Sinnvoll für Systeme welche nahe beieinander liegen und sehr kurze laufzeiten haben. Bei langlaufenden Transaktion kann dieses Verhalten zu unnötigen Blockaden führen. Lösung „Compensation oder Long-Running-Transactions

2-Phase-Commit: Hohe Kopplung
Compensation: Lose Kopplung

8 XML-Technologien für die Integration

8a) Begründen Sie die Notwendigkeit und den Aufbau/Idee des Standards „XML-Encryption". Erläutern Sie, warum ist eine Verschlüsselung auf Ebene der Transport-Layer nicht ausreichend ist.

- Verschüsselung von Nachrichten auf ebene von XML
- Es kann die ganze Nachricht oder Teile verschlüsselt werden
- Es können symmetische und assymetrische Verfahren verwendet werden. Basiert auf Standard-Kryptogrphie-Verfahren
- TA nicht ausreichend das es nur eine punkt-zu-punkt und keine End-To-End-Verschlüsselung ist.

8b) Geben Sie ein sinnvolles Einsatzbeispiel für den Standard XML-Signature an.

Sicherstellen von Veränderungen des Nachrichteninhalts, obwohl die Nachricht lesbar übertragen wird

8c) Warum ist es notwendig XML-Dokument vor der Berechnung einer Signatur zu „kanonisieren/normalisieren"?

Inhaltlich identische Dokumente können sich aufbgrund der Formatierung unterscheiden. (z.B. whitespaces, Reihenfolge der Attribute)

Damit für inhaltsgleiche Dokumente die gleiche Signatur berechnet wird werden diese vor Verwendung normalisiert.

---------- ENDE DER PRÜFUNG ----------