7437 - EDI und E-Business Standards

Electronic
Data
Interchange
(Elektronischer Datenaustausch)

Klassisches EDI - der Kern

Einleitung - die Kernkomponenten
File Transfer- und Messaging-Standards
UN/EDIFACT und EANCOM im Detail
Applikationsschnittstellen
Konverter- und Mappingtechniken

EDI: Die Kernkomponenten

- ✓ EDI-Standardaustauschformate
- **Applikationsschnittstellen**
- **Mapping**
- ✓ Routing
- ✓ Messaging / File Transfer
- **Extras**
  - Archivierung
  - Reporting
  - Alarmierung
  - Tracking & Tracing

Applikationsschnittstellen

Schnittstellenarten

Die SAP IDoc-Schnittstelle - ein "prominentes" Beispiel

Methoden der Inhouse-Anbindung

- **Dateischnittstellen**
  - die traditionelle Methode
- Austausch von Speicherstrukturen durch **IPC-Verfahren**
  - für zeitkritische Anforderungen
- Direkte Kopplung über **Datenbankzugriffe** auf Applikationstabellen
  - nur in speziellen Umgebungen geeignet
  - risikoreich aus Applikationssicht
  - nicht gerade modular
  - elegant, wo der Einsatz vertretbar ist

Dateischnittstellen: Datenformat-Arten

- *Fixed record*-Strukturen
  - Positiv:
    - einfach
    - schnell in der Verarbeitung
    - gut zu parsen
  - Negativ:
    - unflexibel bei späteren Anwendungen
    - verschwendet viel Platz
  - Beispiele:

```
01      12345 20040629...
02 10 00000123 4 10...
```

    - SAP IDocs; SEDAS, GENCOD

Dateischnittstellen: Datenformat-Arten

- *Variable* record-Strukturen
  - Positiv:
    - Kompakt
    - Flexibel bei Feldlängenänderungen
  - Negativ:
    - Trennzeichen erforderlich (verkomplizieren das Parsen: ESC-Mechanismus erforderlich (ESC = "*escape*"))
    - Nur seriell verarbeitbar (!)
  - Beispiel:
    - CSV (*comma separated variables*):

```
01;12345;20040629;...
02;10;1234;10;...
```


Dateischnittstellen: Datenformat-Arten

- Sonstige: *key/value*-Listen
  - Auch "*stanzas*" bzw. Strukturen wie Windows *.ini-Dateien
  - Positiv:
    - sehr flexibel, auch bei Aufnahme neuer Felder
    - selbst-dokumentierend (über sprechende *keys*)
  - Negativ:
    - Overhead
    - kein *a priori*-Wissen über den Ort eines erwarteten Wertes
  - Beispiel:

```
recID=header,orderNo=12345,orderDate=20040629, …
recID=item,matNo=1234,quantity=10,...
```

Dateischnittstellen: Datenformat-Arten

- Sonstige: *Markup*, insb. XML
  - Positiv:
    - ideal für hierarchische *record*-Strukturen
    - selbst-dokumentierend
      - in einfacher Form über sprechende *tags*
      - ggf. auch detailliert, über DTD bzw. Schema
    - validierbar
    - Maßgeschneiderte Datentypen möglich (per XML Schema)
  - Negativ:
    - sehr großer *overhead*
    - hoher Speicherverbrauch - massendatentauglich?
    - komplexes Interface (DOM)

Dateischnittstellen: Datenformat-Arten

- Beispiel zu XML (fiktiv)

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<DOCTYPE my_orders ...>
<my_orders>
  <header>
    <order>
      <refno>12345</refno>
      <date fmt="...">20040629</date>
    </order> <!-- ... -->
  </header>
  <items>
    <item no="10">
      <matNo>1234</matno>
      <quantity unit="pieces">10</quantity>
    </item> <!-- etc. -->
  </items>
<my_orders>
```

Dateischnittstellen: Datenformat-Arten

- Bemerkungen zum Begriff ***flat file***
  - Vorsicht - keine einheitliche Verwendung!
  - Zwei recht konträre, aber gebräuchliche Bedeutungen:

    a) serialisierte Speicherstruktur eines komplexen Typs
    (etwa: "flachgeklopfte Hierarchie")

    b) 1:1-Abbildung des EDIFACT *interchange*
    als *fixed record*-Format, 1 *record* pro Segment

Die SAP EDI-Schnittstelle- ein “prominentes” Beispiel

Das IDoc-Konzept
IDoc-Struktur
IDoc-Verwaltung
Besonderheiten, Ausblick

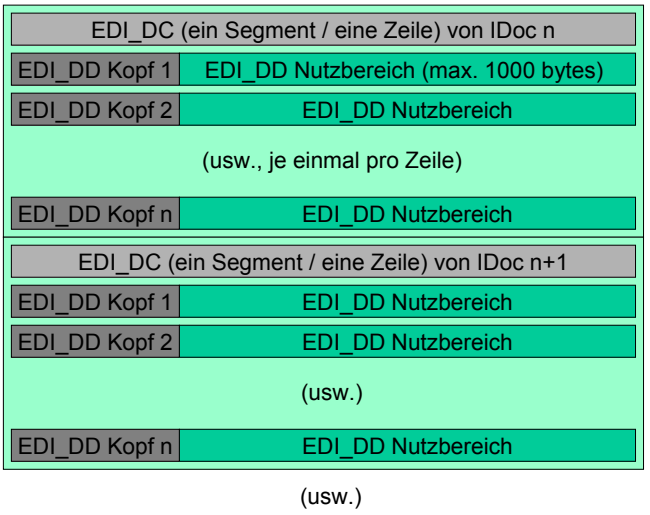
Das IDoc-Konzept

- IDoc = *intermediate document* = Zwischenbeleg
- Eine Struktur...
  - zwischen Anwendungsbeleg einerseits
  - und klassischem, sehr dem EDI-Format nachempfundenen "*flat file*“
- Kernidee:
  - Abstraktionsschicht oberhalb konkreter EDI-Standards wie UN/EDIFACT oder ANSI X12
- Einheitliche Kommunikationsform zwischen
  - verschiedenen SAP-Systemen bzw.
  - SAP und sog. “Sub-Systemen”, insb. EDI-Servern

IDoc-Struktur

- **EDI_DC**
  - Allgemeiner Verwaltungskopfsatz
  - In etwa analog zu einer Kombination aus UNH und UNB
  - Geht jedem IDoc genau einmal voraus
- **EDI_DD**
  - Verwaltungsrahmen für Nutzsegmente
  - Nutzsegmente (je max. 1000 Bytes):
    - mit Versionsnummer
    - fester Satzaufbau
    - Hierarchie
    - Erweiterbarkeit
- **EDI_DS**
  - Statussatz, für Rückmeldungen ausgehender Belege

Struktur einer IDoc-Datei, schematisch



IDoc-Struktur

- XML-IDocs
  - Gleicher Inhalt, neue Verpackung
  - Eher noch größeres Datenvolumen als bei den bereits recht großen IDocs mit *fixed record*-Struktur
- **IDoc-Dokumentation** - *online*-Beispiele
  - HTML-Dokumentation, wie man sie aus SAP R/3 exportieren kann
  - a) EDI_DC, EDI_DD, EDI_DS
  - b) Nutzdaten-Beispiel: ZINVOIC1

IDoc-Verwaltung

- Übergabetabellen (DB)
- Eigene Statusverwaltung der IDocs
  - intern wie extern
  - mit eigenem *user interface*
- Dokumentation in IDoc-Verwaltung integriert
  - IDoc-Formatbeschreibungen sind jederzeit exportierbar
    - in strukturierter Form, vergleichbar XML-DTD/Schema
    - als HTML-Dokumentation
    - als „C"-Headerdateien (Makros, *struct*'s, …)
- Technische Schnittstellen
  - EDI (i.w. Dateischnittstelle plus *RFC* (SAP's RPC-Methode)
  - ALE („*application link enabling*") - *high-level* RFC

Die technische EDI-Schnittstelle

- EDI Ports - oder: SAP erreicht das Betriebssystem
  - Verzeichnisstruktur:
    - Basis-V. z.B. **`/usr/sap/edi/<SAPSYS>/<PORTNAME>`**
    - **`./in`** für eintreffende IDoc-Dateien
    - **`./out`** für ausgehende IDocs
    - **`./status`** für eingehende Dateien mit Statussätzen
  - Ausgehende Daten: **rfcexec**
    Nahtstelle zwischen SAP RFC und Betriebssystem
    - Polling vs. Triggern für jedes IDoc vs. Trigger pro Exportdatei
    - (Vorsicht - ggf. splitten)
  - Eingehende Daten und Statussätze: **startrfc**
    - Polling (durch SAP) vs. Triggern
    - Problem Fehlerbehandlung:
      Welches IDoc ist betroffen, welche sind schon verarbeitet?

Die technische EDI-Schnittstelle

- ALE Ports
  - Austausch von Speicherstrukturen direkt über ein Netzwerkprotokoll, ohne Dateisystemkontakt
  - Erfordert ALE-Fähigkeit auf beiden Seiten
  - Gut insbesondere für zeitkritische Anwendungen
  - ALE wird insbesondere zum Belegaustausch zwischen SAP-Systemen verwendet, seltener zwischen SAP und Subsystemen wie EDI-Servern.
  - Strukturanpassungen notwendig, falls Kopplung zwischen SAP-Systemen mit unterschiedlichen *release*s,
    - etwa R/3 4.6 zu koppeln mit R/3 3.1

Die technische EDI-Schnittstelle

- Besonderheiten
  - Bildung von Applikationsserver-Gruppen
    - **`startrfc`** wendet sich an einen *messaging server* (am besten auf dem ohnehin stets verfügbaren DB-Server betrieben),
    - dieser bestimmt Servermitglied aus "EDI-Gruppe" für den Import
  - Vorteile
    - Lastverteilung
    - Risikostreuung
    - Permanente Verfügbarkeit der EDI-Schnittstelle auch wenn ein Applikationsserver gewartet wird

Die technische EDI-Schnittstelle

- Ausblick / Verwandte Themen
  - BAPIs (Business APIs)
    - genormte *high-level* Schnittstellen z.B. zur Erzeugung eines kompletten Geschäftsdokuments durch ein externes System
  - Business Integration Server (MOM, XML, *Web services*)
  - SAP Business Connector
    - Kostenlose Zusatzkomponente, Einstiegshilfe in XML-Tech / WS
    - Reduzierte Version von einem Produkt von WebMethods
  - SAP NetWeaver
    - SAP's neue eigene Integrations-Technologie
    - SOAP-basierte Web Services bilden eine wesentliche Grundlage

Mapping und *Mapping*-Techniken

Das Grundproblem

Das *Metamap*-Konzept

Andere Ansätze

Spezielle *Mapping*-Aufgaben

Werkzeuge und methodische Ansätze

Zum Begriff „*Mapping*“

- engl. „*map*“ = „Abbildung“ (math.)
- Herstellung / Beschreibung des Zusammenhangs zwischen eingehenden und ausgehenden Daten eines EDI-Konverters, i.d.R. auf Belegebene. Beispiel:
  - Eine SAP IDoc-Struktur „ZINVOIC1“ umwandeln in eine EANCOM-Nachricht „INVOIC“
- Der Begriff suggeriert eine einfache 1:1-Zuordnung zwischen Feldinhalten auf Quell- und Zielstrukturseite, beschreibbar durch eine Zuordnungsvorschrift in Tabellenform
- Tatsächlich können derartige I/O-Beziehungen recht komplex werden. Sie sind dann besser algorithmisch als tabellarisch beschreibbar.

Mapping: Das Grundproblem

- Wunsch und Wirklichkeit
  - Die Anwendung möchte oder kann nur Daten in immer gleicher Weise bereitstellen bzw. akzeptieren.
  - Geschäftspartner benutzen zwar EDI-Standards, aber diese Standards lassen noch eine Fülle von partnerspezifischen Variationen zu.
- Beispiel für Variationen trotz Standardisierung
  - Die Tabelle der Partnerspezifika (ca. 100 x 8 !) aus den Anwendungsrichtlinien des AK Handel zu Rechnungsdaten
- Mapping-Techniken
  - Je nach Vielfalt der Anforderungen ist die passende *Mapping*-Technik (und damit u.U. das dazu passendste Konverterprodukt) zu wählen.

Das Metamap-Konzept

- **Algorithmischer Grenzfall:** *Mapping*=Software-Entwicklung
- Gemeinsamer, komplexer Programmcode
  - für viele Partnerspezifika
  - mit vielen Fallunterscheidungen / Logikabschnitten
  - redundanzarm, leicht zu warten bei gemeinsamen Veränderungen
- Softwareschalter und optionale Parameter
  - Steuerung im Einzelfall
  - Zusammenfassung zu **Schalter-Sets**
  - Organisation z.B. als Lookup-Tabelle:
    - Eine Zeile pro Schaltername, eine Spalte pro Set
- Auswahl des zuständigen Schalter-Sets mittels Absender/Empfängerkennung
  - z.B. aus UNB-Segment oder EDI_DC

[Quelle: Auch Mapping ist Software-Entwicklung. H.Werntges, edi-change 02/2001, S. 27 - 30]

Das Metamap-Konzept

- Redundanzvermeidung der 2. Ordnung:
  - Funktionale Schaltergruppen.
  - Beispiel Netto-vs. Bruttoabrechnungsverfahren
    - Eine Gruppe repräsentiert „Netto“, eine andere „Brutto“
  - Schalter-Sets
    - Komplette Schaltersammlung
    - Gruppierung: Jeweils ein Set für alle Partnerkennungen mit gleichen Anforderungen
  - Vererbungskonzept:
    - Sets erben Werte von Schaltergruppen und anderen Sets
    - Geerbte Werte können bei Bedarf lokal überschrieben werden
    - Beispiele:
      - Set X wie Set A, aber mit Nettoabrechnungs-Gruppe
      - Set B wie Set A, zusätzlich Schalter S := „M“

Das Metamap-Konzept

- Redundanzvermeidung der 2. Ordnung (Forts.):
  - Ergebnis: Redundanzen der Schaltermatrix minimiert
  - Voraussetzungen dazu
    - Vererbungstechnik
    - Default-Sets
    - Erst Fachwissen über die Bedeutung der Schalter gestattet sinnvolle Definition von Schaltergruppen
    - Analyse der Schalterbeziehungen und insbesondere die Definition des Vererbungsgraphen erfordert Erfahrung, Branchenkenntnis und Vorausschau

- **NAD-Mapping** - ein separater Einsatz für Metamaps
  - Partnerfunktionen (z.B. „Besteller“, „Rechnungsempfänger“) aus Sicht des EDI-Partners unterscheiden sich oft von der eigenen Repräsentation in den Stammdaten, also:
  - Partnerspezifische Anpassungen notwendig

Mapping: Andere Ansätze

- **Zweistufiges *Mapping***
  - (1) Quelle -> interne Struktur, (2) interne Struktur -> Zielformat
  - Variante: Semantische Geschäftsprozess-Integration
    - interne Struktur erzwingt semantisch korrekte Bereitstellung
    - zweiter Teil des *Mapping* ist dann frei von Missverständnissen

- Der andere Grenzfall: ***Mapping* - wörtlich genommen**
  - Viele, aber einfache tabellarische *Maps*
  - Vorteile:
    - Keine Programmierlogik, GUIs einsetzbar,
    - *Mapping* nach Spezifikation durch angelernte Kräfte möglich
  - Nachteile:
    - Schlecht wartbar
    - Aufwändig bei systematischen Änderungen

Spezielle *Mapping*-Aufgaben

- **Plausi-Checks**
  - wenn möglich auch automatische Korrekturverfahren
  - Bsp: Materialstamm; Rechnungslisten
- **Nachrüsten** (konstanter) Stammdaten
  - Bsp: Metro Lieferantennr.
- **Ausnahmetabellen**
  - Bsp. spezielle Materialien
- **Logs** für Warnungen
  - Bsp. EAN-Lücken, ILN-Lücken
- Erzeugen von **Meta-Belegen**
  - Bsp. Sammel-Rechnungsliste für REWE
- Während der Entwicklung:
  - **Test-Suite** !

Spezielle *Mapping*-Aufgaben

- Angepasste Fehlerbehandlung
  - *en bloc*
    - alle Belege ablehnen bei einem Fehler
    - notwendig bei Abhängigkeiten zwischen Belegen,
      - z.B. bei einem Bündel Rechnungen mit Rechnungsliste
  - *split*
    - Nur fehlerhafte Belege abtrennen,
    - Rest konvertieren & versenden
    - Verschafft Zeit zur Fehlerkorrektur
    - Hält nur die Problemfälle zurück

Grundsatzfrage beim *Mapping*

- Welche Logik / Funktion gehört in den Konverter, welche in die Anwendung?

- Empfehlung:
  - Anwendung:
    - Gemeinsame Funktionalitäten
    - generelle Logik
  - Konverter:
    - Partnerspezifika

Werkzeuge / methodische Ansätze

- Grenzfall 1: Mapping wörtlich genommen
  - Felder möglichst 1:1 abbilden, Programmierlogik vermeiden
  - Graphische Mapping-Werkzeuge evtl. gut einsetzbar

- Grenzfall 2: Große Vielfalt, komplexe Regeln, algorithmisch beschreibbar
  - Klassische Software-Entwicklung mit geeigneter *Mapping*-Sprache

- Technische Umsetzung in Konverterprodukten
  - Interpreter-Ansatz (typisch) vs. kompilierter Code
  - Herstellerabhängig
  - Allgemein: *Tradeoff* zu suchen zwischen
    - einfach zu erlernenden und bedienenden Werkzeugen, sowie
    - flexiblen, günstig zu wartenden, performanten und skalierbaren Umgebungen

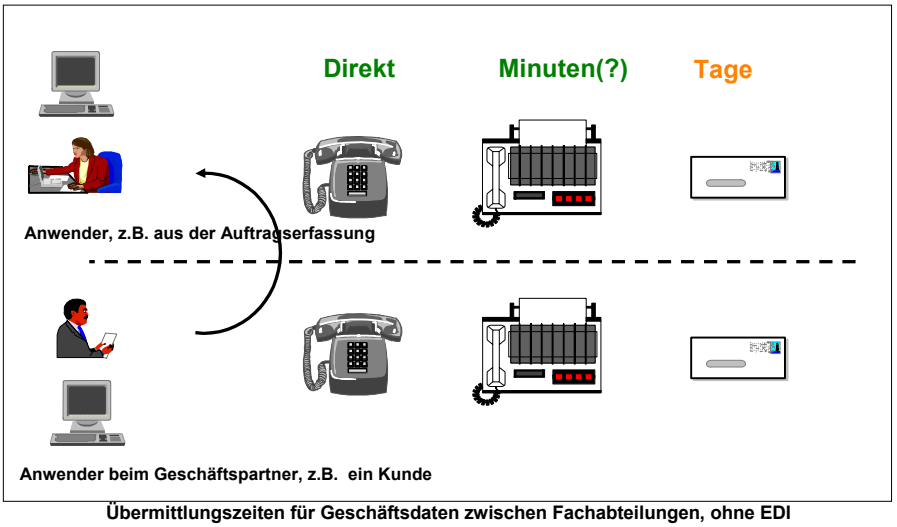
Das Umfeld

Tracking, Reporting, Archivierung
Hybride Ansätze
EAI und BPI

EDI Tracking & Tracing (EDI T&T)

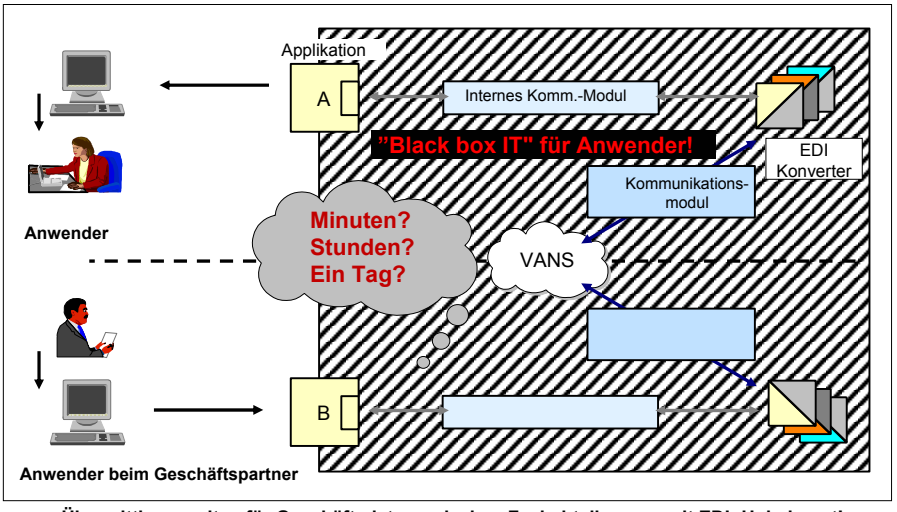
- Material:
  - Einleitung: Folgende 4 Folien
  - Zeigen: PDF-Dokument (anstelle On-line Demo) mit Screenshots zu EDI Tracking & Tracing, incl. integrierter Archivanbindung, Reporting-Funktionen und aktiven Steuerungsmöglichkeiten.
- Quelle:
  - EDI Tracking & Tracing. H. Werntges, edi-change 3/2001, S. 26-34

Tracking, Reporting, Archivierung



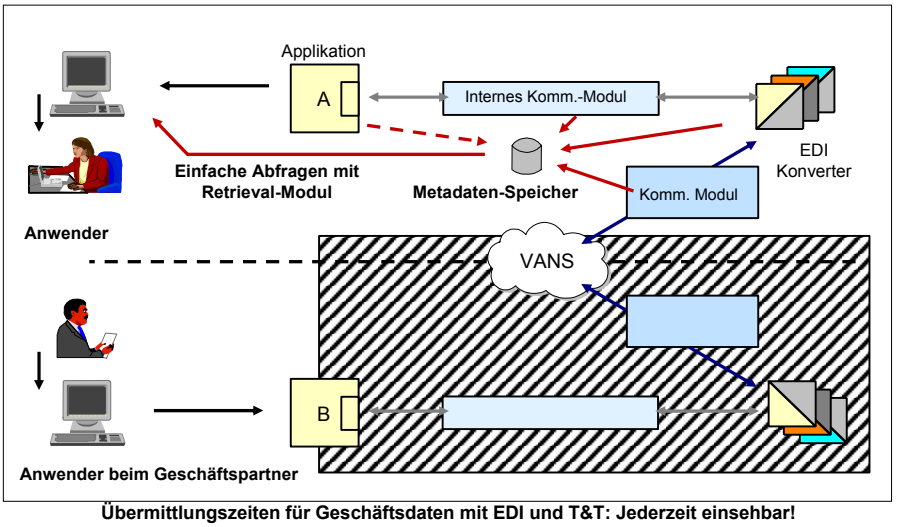
Übermittlungszeiten für Geschäftsdaten zwischen Fachabteilungen, ohne EDI

Tracking, Reporting, Archivierung



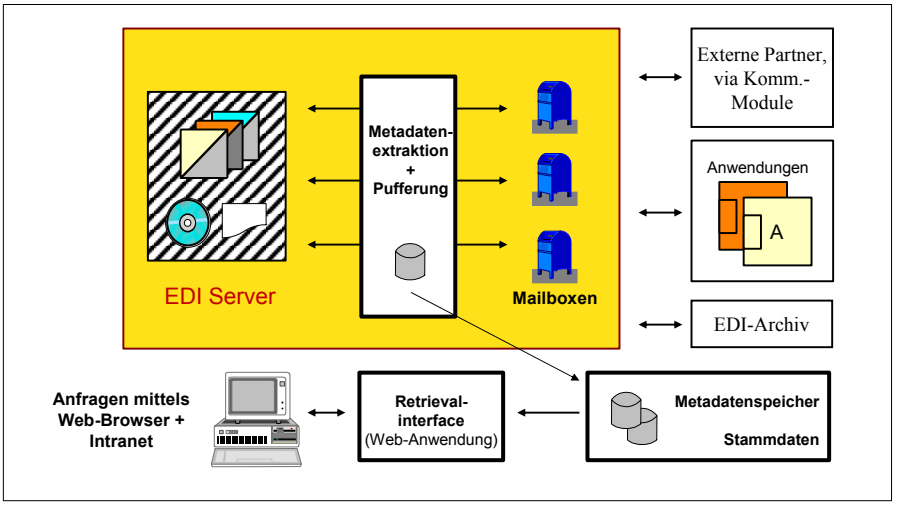
Übermittlungszeiten für Geschäftsdaten zwischen Fachabteilungen, mit EDI: Unbekannt!

Tracking, Reporting, Archivierung



Übermittlungszeiten für Geschäftsdaten mit EDI und T&T: Jederzeit einsehbar!

Tracking, Reporting, Archivierung



Komponenten von EDI Tracking & Tracing

Hybride Ansätze

- Zum Lieferanten: **Web-EDI**
- Zum Kunden: **Web-Shops**
  - **Beispiele: CAIPRO (Karstadt/Quelle), Metro-EDI, ...**
- EDI-to-paper: **ePost**
- FAX-to-EDI:
  - OCR, KI-Methoden, plus manuelle Nachkontrolle (hier nur Stichworte)
- Optional: Vertiefung der Einzelthemen
  - z.B. per Referat

EAI und BPI

- **EAI**: *Enterprise Application Integration*
  - Einheitliche Infrastruktur / APIs zur Erleichterung von Kopplungen zwischen diversen Anwendungen („A2A“)
  - Konverter können Bestandteile einer Gesamtlösung sein
  - Zentrale Administrierbarkeit spielt eine große Rolle
  - Sinnvoll für große Unternehmen mit einer Vielzahl von Anwendungen an unterschiedlichen Standorten
  - Vgl. 4 Abbildungen aus edi-change 02/2001
- **BPI**: *Business Process Integration*
  - Baut auf EAI auf
  - Geschäftsprozesse statt Einzelschritte im Vordergrund
  - Neu: Auch Integration manueller Schritte; *workflows*

EAI: Die Klassiker

- *Hub-and-spoke* Ansatz
  - z.B. GE Enterprise
  - siehe auch JMS
- *Message queues*
  - insbesondere IBM MQ Series
  - *point-to-point queues*, damit flexible Topologien realisierbar
- TIBCO, oder: *The Information Bus company*
  - Lokale „Konnektoren“, gemeinsamer „info bus“
  - Zentral administrierbar, sehr gut für *publish/subscribe*-Verfahren

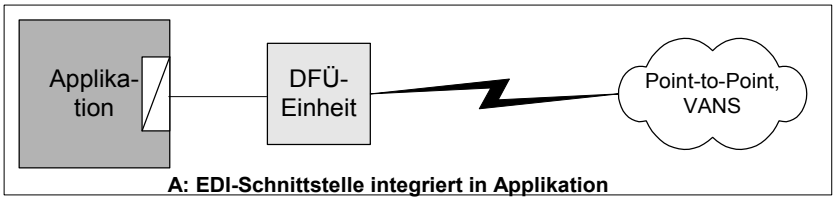
EAI: Neuere Entwicklungen

- Herstellerneutrales API:
  - *Java Message Service* (JMS, aus J2EE)
- Infrastruktur hinter dem API im Prinzip austauschbar
  - Beispiel: SonicMQ, lieferte Lösung für CommerceOne-Anbindung
- Sowohl für P2P als auch für P/S geeignet
- Proprietäre Gesamtkonzepte
  - SAP: NetWeaver
  - Oracle: J2EE-basiert
  - IBM: WebSphere

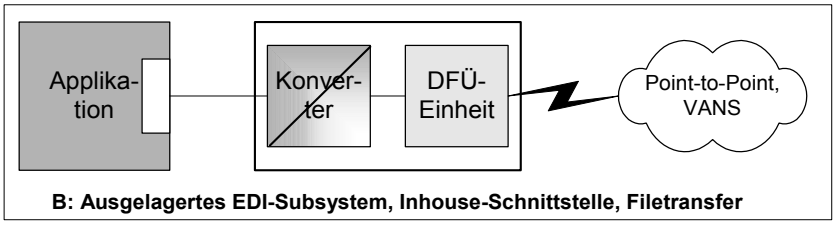
EAI: Zielkonflikte

- Einerseits
  - Hoher Durchsatz
  - Eignung für viele kleine Nachrichten (kB-Bereich, auch SMS)
- Andererseits
  - *Heavy duty file transfer*: GB-Bereich
- *Multicasting*-artige Verteilung
  - z.B. Börsenticker-Angaben, per *publish/subscribe*-Verfahren
  - für flüchtige, kleine, sich ständig aktualisierende Nachrichten
- aber auch:
  - Sichere P2P-Übertragung von Dateien mit Zustell-Bestätigung und *Tracing*.
- Fazit:
  Auswahl einer EAI-Lösung erfordert Bedarfsanalyse.

EDI/EAI: Historische Entwicklung

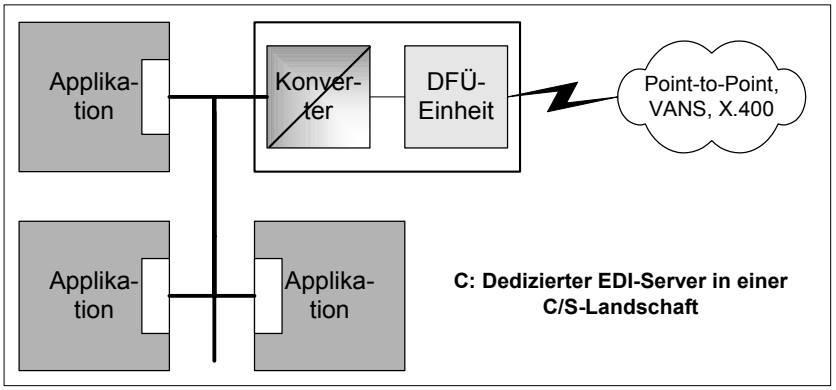


A: EDI-Schnittstelle integriert in Applikation



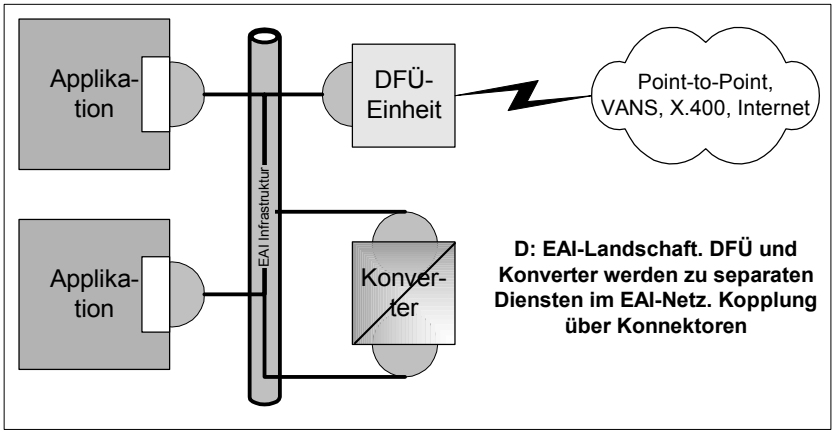
B: Ausgelagertes EDI-Subsystem, Inhouse-Schnittstelle, Filetransfer

EDI/EAI: Historische Entwicklung



C: Dedizierter EDI-Server in einer C/S-Landschaft

EDI/EAI: Historische Entwicklung



D: EAI-Landschaft. DFÜ und Konverter werden zu separaten Diensten im EAI-Netz. Kopplung über Konnektoren