Architekturen von Informationssystemen -

von Handels-H-Modellen, Workflow-basierten PPS-Systemen und Client-/Server-Konzepten

Jörg Becker, Holger Hansmann, Tobias Rieke

Institut für Wirtschaftsinformatik
Westfälische Wilhelms-Universität
Leonardo-Campus 3
48149 Münster
{becker|ishoha|istori}@wi.uni-muenster.de

Abstract: Informationsmodelle und Informationssystem-Architekturen besitzen in der betriebswirtschaftlichen Forschung und Praxis mittlerweile einen sehr großen Stellenwert. Exemplarisch werden in diesem Beitrag der Handels-H-Ordnungsrahmen als Vertreter einer fachkonzeptionellen, domänenspezifischen Architektur, eine Workflow-basierte PPS-Architektur als Vertreter einer sowohl fach- als auch DV-konzeptionellen, domänenspezifischen Architektur- und Mehr-Schichten-Konzeptionen als Vertreter von domänenneutralen, DV-spezifischen Architekturen vorgestellt.

1 Informationsmodelle und Informationssystem-Architekturen

Als Hilfsmittel für die Erklärung und Gestaltung von Informationssystemen werden Informationsmodelle eingesetzt [BS03, SNZ95]. Ein *Informationsmodell* ist die Repräsentation eines betriebswirtschaftlich relevanten Sachverhaltes für die Zwecke der Organisations- und Anwendungssystemgestaltung [Be01]. Eine *Informationssystem-Architektur* ist ein spezielles Informationsmodell auf einem hohen Abstraktionsniveau, das die Bestandteile eines Informationssystems und ihre Beziehungen zueinander verdeutlicht und ein didaktisches und gedankliches Hilfsmittel für die Beschreibung und Einordnung von Informationssystemen darstellt [BS03, Kr90, Te99]. Informationssystem-Architekturen, die auch als *Ordnungsrahmen* bezeichnet werden, [BM03, Me01] können unabhängig von einer betrieblichen Anwendungsdomäne oder domänenspezifisch sein. Domänenneutrale Architekturen dienen einer allgemeingültigen Klassifikation von Komponenten beliebiger Informationssysteme (z. B. ARIS [Sc01]), während domänenspezifische Architekturen Informationssysteme repräsentieren, welche die spezifischen Anforderungen einer betrieblichen Anwendungsdomäne berücksichtigen (z. B. Architekturen für Warenwirtschaftssysteme) [Be03, Ne03, Ha03]. Das Handels-H-Modell [BS03] stellt beispielsweise eine domänenspezifische Architektur für Handelsun-

ternehmen dar, während das Y-CIM-Modell [Sc97] auf Industrieunternehmen fokussiert. Neben der Unterscheidung in domänenspezifische und domänenneutrale Architekturen lässt sich auch eine Einteilung in die fachkonzeptionelle oder DV-konzeptionelle Ausrichtung der Architektur vornehmen. Während das Fachkonzept eine enge Bindung zur betriebswirtschaftlichen Problemstellung besitzt, fokussiert das DV-Konzept auf die Übertragung der Begriffswelt des Fachkonzepts in Kategorien der DV-Umsetzung [Sc97]. Im Folgenden werden Beispiele für domänenneutrale und domänenspezifische sowie für fach- und DV-konzeptionelle Architekturen erläutert.

- *Handels-H*: Domänenspezifische Architektur, die betriebswirtschaftliche Anforderungen an Handelsinformationssysteme auf fachkonzeptioneller Ebene beschreibt.

- *Architekturen Workflow-basierter PPS-Systeme*: domänenspezifische Architektur, die sowohl eine fachlich-semantische Integration der Aufgaben zur Prozesskoordination in der PPS als auch die DV-konzeptionelle Integration von PPS- und Workflow-Komponenten spezifiziert.

- *Mehr-Schichten-Architekturen*: domänenneutrale Architekturen, die die Aufgabenteilung verschiedener Architekturschichten von Anwendungssystemen auf DV-konzeptioneller Ebene beschreiben.

Die Festlegung der Architekturen in unterschiedlichen Bereichen gehört zu den zentralen Aufgaben eines Informationsmanagers im Unternehmen. Inhaltlich, methodisch und technisch gibt er mit den Architekturen die Leitlinien vor, an denen sich die Gestaltung der Informationssysteme (einschließlich des Erwerbs von Standard-Anwendungssoftware) auszurichten hat.

2 Handels-H-Architektur

Das Handels-H stellt eine domänenspezifische, fachkonzeptionelle Architektur eines Informationssystems für den Handel dar (vgl. im Folgenden Abb. 1 sowie [Bs03]). Dazu grenzt es die Teilsysteme eines Handelsunternehmens ab und positioniert sie hinsichtlich ihres inhaltlichen Zusammenhangs. Im Handels-H sind die zum Beschaffungsbereich gehörenden Teilsysteme Einkauf, Disposition, Wareneingang, Rechnungsprüfung, Debitorenbuchhaltung und die zum Vertriebsbereich gehörenden Teilsysteme Marketing, Verkauf, Warenausgang, Fakturierung und Debitorenbuchhaltung durch das Lager, welches die Überbrückungsfunktion zwischen Beschaffung und Distribution ausübt, gekoppelt. Die betriebswirtschaftlich-administrativen Aufgaben Haupt- und Anlagenbuchhaltung, Kostenrechnung und Personalwirtschaft bilden das „Fundament“ des Handels-H. Das „Dach“ umfasst die Bereiche Controlling, Executive Information Systeme (EIS) und Systeme zur Unterstützung der Unternehmensplanung und stellt den Entscheidungsträgern verdichtete Informationen zur Führung des Handelsunternehmens bereit.

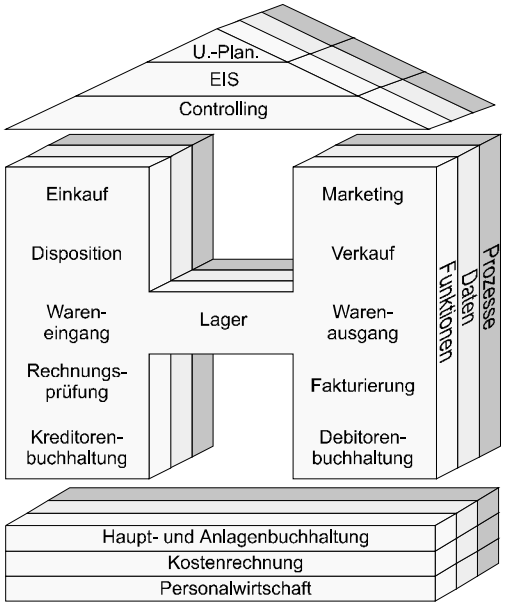

Abb. 1: Das Handels-H-Modell [BS03]

Ausgehend von diesem Ordnungsrahmen, der die Teilsysteme des Handels spezifiziert und hinsichtlich ihres inhaltlichen Zusammenhangs positioniert, bietet er die Möglichkeit einer Navigation durch die den einzelnen Teilsystemen zugeordneten Referenzmodellen. Die einzelnen Referenzmodelle lassen sich jeweils in eine von drei Beschreibungssichten auf das Handel-H (Funktionen, Daten und Prozesse) einordnen. Schnittstellen der Referenzmodelle verbinden sie innerhalb und zwischen den Teilsystemen und Sichten. Eine Organisationssicht, wie sie im ARIS-Modell [Sc97] vorhanden ist, findet keinen Eingang in das Handels-H, da sie nur unternehmensindividuell mit Bezug auf vielfältige Umweltfaktoren (z. B. Angebotsprogramm, Internationalisierung, Organisationsgröße etc.) vorgenommen werden kann. Die einzelnen Objekte (z. B. Funktionen, Ereignisse, Entity-Typen etc.) werden einmalig definiert und in den einzelnen Modellen mehrfach verwendet. Damit wird aus Gründen der Übersichtlichkeit mehreren Teilmodellen mit redundanten Objekten der Vortritt vor einem redundanzfreien Gesamtmodell gegeben. Auch können die Referenzmodelle über die Darstellung in logisch abgeschlossenen Einheiten eindeutig einem Teilsystem zugeordnet werden. In Abb. 2 ist exemplarisch das Prozessmodell „Einlagerung der Ware“ aus dem Teilsystem Wareneingang dargestellt. Es enthält Schnittstellen zum Prozess „Warenbewertung“ des Wareneingangs, „Abnehmerretoure“ des Warenausgangs und „Umlagerung“ des Lagers. Ferner stellen die Ereignisse „Ware ist einzulagern“ aus dem Prozessmodell „Retoure an den Lieferanten“ und „MTV-Abwicklung ist durchgeführt“ aus dem Prozessmodell „Wareneingang Lager“ Verbindungen zu weiteren Prozessmodellen her (redundante Darstellung der Ereignisse).

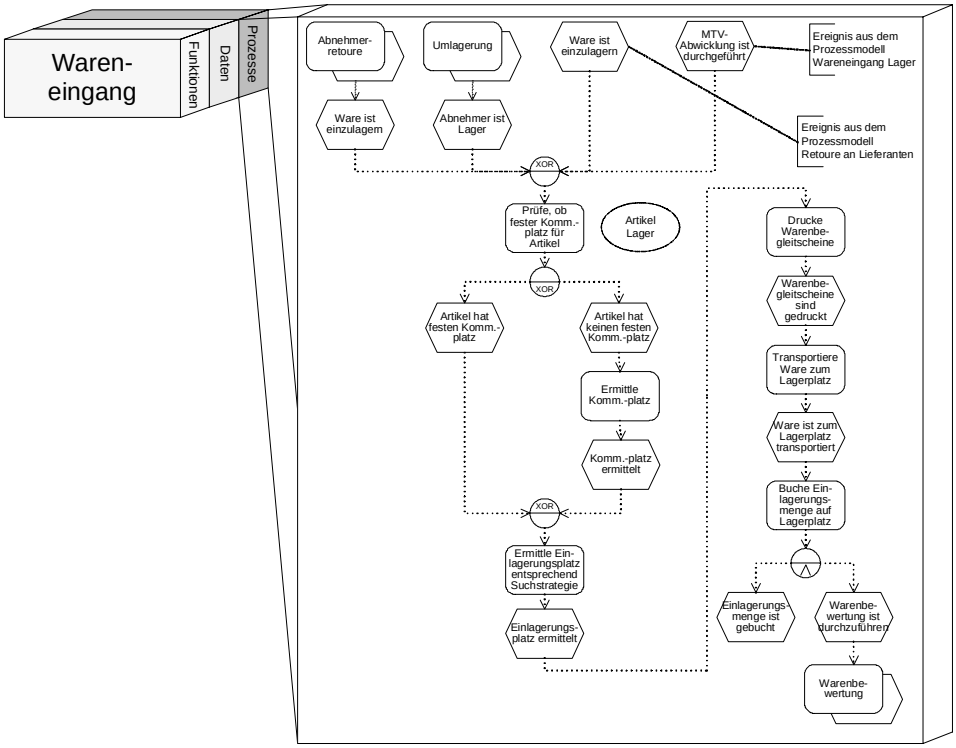

Abb. 2: Prozessmodell „Einlagerung der Ware“ [BS03]

Neben der in Abb. 1 dargestellten Architektur für das Lagergeschäft umfasst das Handels-H u. a. auch Modelle für den Streckenprozess und das Zentralregulierungsgeschäft. Beim *Streckenprozess* entfallen die Teilsysteme Wareneingang, Lager und Warenausgang vollständig (vgl. Abb. 3), da hier die Ware direkt vom Lieferanten zum Abnehmer gelangt, während der dispositive Informationsfluss (Auftrag und Bestellung) und der Wertefluss (Abnehmerrechnung und –zahlung resp. Lieferantenrechnung und –zahlung) sich weiterhin zwischen Kunde und Handelsunternehmen resp. Handelsunternehmen und Lieferant bewegen. Das *Zentralregulierungsgeschäft* findet sich vornehmlich bei Einkaufsgemeinschaften und gelegentlich bei Großhändlern. Innerhalb der Zentralregulierung fokussiert sich die Leistung auf die finanzwirtschaftliche Abwicklung des Handels, indem das Handelsunternehmen zentral die Regulierung und optional das Delkredere für die Kunden übernimmt. Damit reduziert sich das Handels-H neben Einkauf und Marketing auf die zentralen Teilsysteme Rechnungsprüfung, Kreditorenbuchhaltung, Fakturierung und Debitorenbuchhaltung (vgl. Abb. 3).

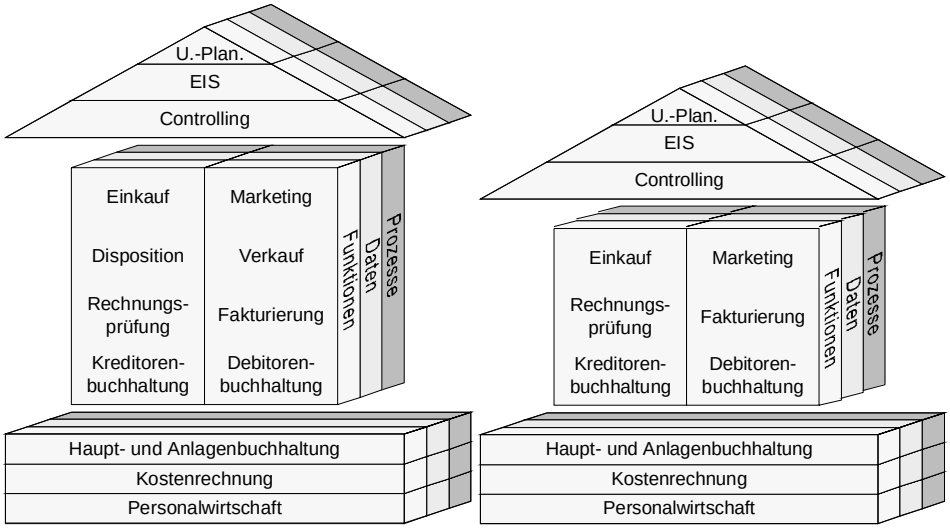

Abb. 3: Geschäftsprozess Strecke und Zentralregulierung [BS03]

3 Architekturen Workflow-basierter PPS-Systeme

Die Produktionsplanung und -steuerung (PPS) hat die Aufgabe, aufgrund erwarteter und/oder vorliegender Kundenaufträge den mengenmäßigen und zeitlichen Produktionsablauf unter Beachtung der verfügbaren Ressourcen durch Planvorgaben festzulegen, diese zu veranlassen sowie zu überwachen und bei Abweichungen steuernde Maßnahmen zu ergreifen, so dass die betrieblichen Ziele erreicht werden [Zä98]. Die PPS stellt durch die Komplexität der Fertigungsprozesse und der erforderlichen Planungs- und Steuerungsverfahren hohe Anforderungen an unterstützende Anwendungssysteme [Ku99, FFG97, LE99].

Trotz der ausgereiften Funktionalität moderner PPS-Systeme und signifikanten Fortschritten in der Datenintegration ist in der PPS noch immer eine Reihe koordinatorischer Defizite wie z. B. mangelnde Anpassbarkeit der Systeme an unternehmensspezifische Planungsprozesse, geringe Reaktionsfähigkeit auf Störungen und mangelnde Prozesstransparenz festzustellen [Ad98, Ku99, BHN02]. Einen Ansatz für die Verbesserung der Prozesskoordination stellen *Workflow-Management-Systeme (WfMS)* dar. WfMS unterstützen die Gestaltung, Steuerung und Kontrolle betrieblicher Prozesse durch die Koordination von Aktivitäten, Anwendungen, Daten und Bearbeitern anhand eines zuvor spezifizierten Prozessmodells. Nutzeneffekte von WfMS sind beispielsweise die Reduktion von Durchlaufzeiten, die Erhöhung der Prozesstransparenz und eine erhöhte Anpassbarkeit Workflow-basierter Informationssysteme an unternehmensspezifische Anforderungen [BV96, GS95, MH03].

Ein kurzer Blick zurück: Die erste Phase der Softwareentwicklung brachte monolithische Systeme hervor, in denen Funktionen, Daten und der Kontrollfluss in einem (COBOL-)

Programm verknüpft waren. Die zweite Phase war durch die Segmentierung in eine separate Daten- und Anwendungsschicht gekennzeichnet, die durch die Entwicklung von Datenbank-Management-Systemen (DBMS) möglich wurde. In Phase drei führten WfMS zu einer *Trennung der Prozesslogik* von der Anwendungs- und Datenschicht (vgl. Abb. 4), d. h. die Steuerungsfunktionalität wurde nach „außen“ in ein WfMS verlagert. Damit trugen sie den steigenden Anforderungen an die Anpassbarkeit von Standardsoftware Rechnung, denn in Verbindung mit der zunehmenden Modularisierung von Anwendungssystemen können WfMS die Rolle einer Integrationsschicht (Middleware) einnehmen, die unabhängige Anwendungssystem-Komponenten entlang der Prozesse eines Unternehmens individuell verbindet und koordiniert [Ha03]. Bisher werden WfMS vorwiegend zur automatisierten Koordination von Verwaltungsprozessen (z. B. Reisekostenabrechnung) eingesetzt, neuere Arbeiten haben jedoch gezeigt, dass auch PPS-Prozesse eine hohe Eignung für eine Unterstützung durch WfMS aufweisen [Ha03, LBL02, HNB01, Lo98].

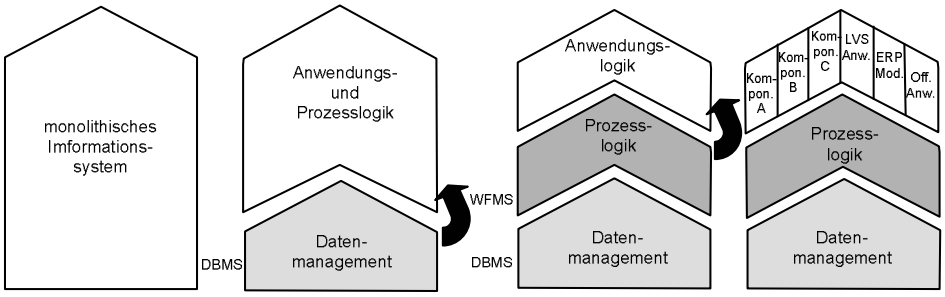

Abb. 4: Trennung der Prozesslogik von Anwendungs- und Datenschicht [MH03]

Die Kopplung von WfMS und PPS-Systemen führt im Normalfall zu einer *Redundanz* hinsichtlich der Koordinationsfunktionalität und der dafür benötigten Daten. Beispielsweise existieren Mechanismen zur Koordination von Aktivitätenfolgen bzw. Kontrollflüssen sowohl in WfMS (Workflowmodell und -instanz) als auch in PPS-Systemen (Arbeitsplan und Fertigungsauftrag). Dies kann zu Konflikten und *Synchronisationsproblemen* bei der Prozesskoordination führen, da die Spezifikation bzw. Konfiguration der Mechanismen der Prozesslogik nicht einheitlich erfolgen kann und Wechselwirkungen zwischen den jeweiligen Mechanismen von WfMS und PPS-System bestehen, die zu Restriktionen für die Ausgestaltung der Prozesskoordination führen. Dies ist beispielsweise der Fall, wenn die Funktion zur Buchung eines Warenausganges im PPS-System so konfiguriert ist, dass sie nach ihrer Beendigung direkt die Folgefunktion zum Erstellen eines internen Transportauftrages anstößt, so dass das WfMS die Koordination der Aktivitätenfolgen nicht ausüben kann und im Workflowmodell für die beiden Aufgaben nicht zwei separate Aktivitäten modelliert werden können [Ha03].

Im Gegensatz zu bestehenden Architekturen, die lediglich eine *technische* Integration von Workflow- und Fachkomponenten realisieren, ist daher auch eine *fachlich-semantische* Integration von Koordinationsmechanismen für die Prozessgestaltung, -steuerung und -kontrolle vorzunehmen, um die beschriebenen Redundanzen und Synchronisationsprobleme zu vermeiden [Ne03]. Es wird somit eine integrierte Gesamtar-

chitektur benötigt, bei deren Konzeption sowohl *fachkonzeptionelle* Anforderungen der betrachteten betrieblichen Anwendungsdomäne (insbesondere die Integration von Aufgaben der Prozesskoordination) als auch *DV-konzeptionelle* Integrationsaspekte (z. B. Schnittstellen für die Integration der aufzurufenden Anwendungssysteme) zu berücksichtigen sind [Ha03].

Den DV-konzeptionellen Anforderungen kann durch den Einsatz geeigneter Schnittstellen Rechnung getragen werden, die bspw. die Standards der Workflow Management Coalition (WfMC) umsetzen. Fachkonzeptionelle Anforderungen wie die Realisierung einer effizienten Koordination unter weitgehender Beseitigung von Koordinationsredundanzen werden von einem *Workflow-basierten PPS-System* erfüllt, das die in Abb. 1 dargestellte domänenspezifische Architektur umsetzt. Diese sieht einen kompletten Neuentwurf der Komponenten eines PPS-Systems und ihrer Aufgabenverteilung vor, da alle zur Workflow-Modellierung, -Steuerung und -Kontrolle konfliktären Koordinationsmechanismen aus den PPS-Fachkomponenten in die *Workflow-Komponenten* verlagert werden. Die Workflow-Komponenten übernehmen somit die Prozessgestaltung, -steuerung und -kontrolle, während *PPS-Fachkomponenten* domänenspezifische Verfahren für die Produktionsplanung realisieren, die auch auf Workflows angewendet werden können. Dies ermöglicht im Gegensatz zu traditionellen Architekturen eine (von PPS-Fachkomponenten durchgeführte) Zeit- und Kapazitätsplanung für Workflowinstanzen. Zusätzlich beinhaltet die integrierte Gesamtarchitektur eines Workflow-basierten PPS-Systems ein *Integrations-Repository*, das Zuordnungen zwischen Workflow- und Fachkomponenten-Daten (z. B. das von einem Workflow bearbeitete PPS-Objekt wie bspw. ein Fertigungsauftrag) verwaltet und auf dessen Datenstrukturen die Workflow- und Fachkomponenten zugreifen.

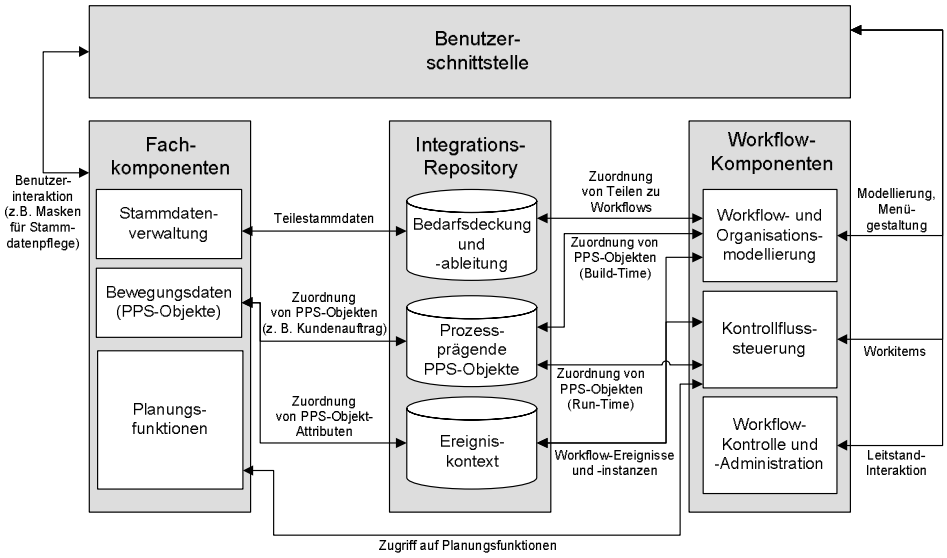

Abb. 5: Referenzarchitektur eines Workflow-basierten PPS-Systems [Ha03]

Die Zusammenführung von Koordinationsmechanismen bedeutet insbesondere, dass *Arbeitspläne* bzw. *Fertigungsaufträge* durch *Workflowmodelle* bzw. -*instanzen* zu ersetzen sind. Dadurch können individuelle Statusdefinitionen für Auftragstypen direkt über Workflow-Ereignisse abgebildet und mithin die Statusübergänge aktiv koordiniert werden. Das WfMS verwaltet die Beziehungen zwischen Produktionsplanungs-Workflows und assoziierten Fertigungsauftrags-Workflows auf Modellebene und zur Laufzeit. Das Monitoring und Manipulieren der laufenden Workflowinstanzen erfolgt durch Verwendung eines so genannten *Workflow-Leitstandes*.

PPS-Objekte wie Teilestamm, Bedarf, Kundenauftrag, Wareneingang oder Bestellung, die prozessprägende Objekte von PPS-Prozessen repräsentieren und nicht als Workflowmodelle abgebildet werden, werden als Stamm- oder Bewegungsdaten in Fachkomponenten verwaltet. Die in traditionellen PPS-Systemen verwalteten *Fertigungsressourcen*, die menschlichen und maschinellen Ressourcen auf verschiedenen Aggregationsebenen entsprechen, sind durch eine Hierarchie von Workflow-Rollen und zugeordneten Workflow-Aktoren (Person oder Maschine) zu ersetzen. Der PPS-Arbeitsvorrat, der eine „To-Do-Liste“ für Disponenten repräsentiert, wird durch eine Workflow-basierte, so genannte Worklist ersetzt. Bei der Rollenauflösung werden die aktuelle Auslastung bzw. die verfügbare Kapazität sowie PPS-Heuristiken wie Prioritätsregeln berücksichtigt.

Erst die Zusammenführung von Koordinationsmechanismen in einer integrierten Gesamtarchitektur ermöglicht in Verbindung mit der Einbindung flexibler, parametrisierbarer Fachkomponenten eine individualisierbare, redundanzfreie und effiziente Prozesskoordination. Ein Workflow-basiertes PPS-System, das der Architektur eines zentrale Koordinationsaufgaben wahrnehmenden WfMS mit angekoppelten Fachkomponenten folgt, leistet somit eine ganzheitliche und durchgängige Gestaltung, Planung, Steuerung und Kontrolle von Prozessen der industriellen Auftragsabwicklung.

4 Mehr-Schichten-Architekturen

Als Entwurfsdimensionen für Informationssystem-Architekturen werden auf der Ebene des DV-Konzeptes der *Verteilungsgrad* (ein zentraler oder mehrere verteilte Server), die *Art der Kopplung* (eng bzw. synchron, lose bzw. asynchron) und die *Schichtung* unterschieden [To02]. Die Schichtung repräsentiert eine Anwendung des Prinzips der Modularisierung [HR89] und wird oftmals als die wesentliche Entwurfsdimension angesehen [Sc93]. Sie dient daher als Gliederungskriterium für die nachfolgenden Abschnitte.

Eine *Architekturschicht* ist eine funktionale Abstraktion, die Dienste für eine übergeordnete Schicht zur Verfügung stellt und mit über- bzw. untergeordneten Schichten über Schnittstellen kommuniziert [No00, Sc93, We02]. Jede Schicht implementiert somit eine abgegrenzte Aufgabe der Gesamtarchitektur (z. B. Datenbankmanagement, Anwendungslogik, Benutzeroberfläche), wobei Anwendungen unterschiedlicher Anbieter für die Realisierung der Dienste auf den einzelnen Schichten eingesetzt werden können. Jede Schicht kann einem separaten Rechner zugeordnet werden (1:1), es können jedoch

auch mehrere Rechner je Schicht (1:n) eingesetzt oder mehrere Schichten auf einem Rechner realisiert werden (n:1) [Ko02, Sa95].

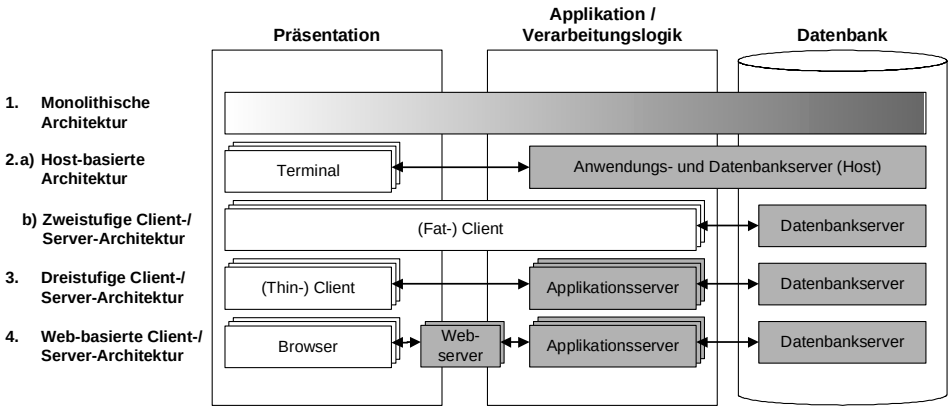

Abb. 6: Altern. Gestaltungsmöglichkeiten für Schichten-Architekturen [vA97, SKM00]

Abb. 6 visualisiert Alternativen für die Gestaltung von Schichten-Architekturen, die zum einen von der *Anzahl der Schichten* und zum anderen von der *Aufteilung von Präsentations-, Verarbeitungs- und Datenbankdiensten* auf die eingesetzten Client- bzw. Server-Anwendungen bzw. -Rechner charakterisiert sind.

4.1 Monolithische bzw. Ein-Schicht-Architektur

Beim Einsatz einer monolithischen Architektur werden alle Schichten durch eine einzige Stand-alone-Anwendung realisiert, die im Normalfall nur auf einem Rechner läuft und daher keinen gleichzeitigen Zugriff durch mehrere Benutzer erlaubt. Die Zugriffe auf Datenstrukturen sowie die Präsentationsdienste werden nicht durch eine separate Schicht gekapselt und standardisiert, sondern durch proprietäre Mechanismen innerhalb der Anwendung realisiert. Eine derartige Systemarchitektur findet sich zwar häufig bei Textverarbeitungen, Programmierumgebungen und Spielen, ist jedoch für den Produktiveinsatz komplexer Systeme wie ERP- bzw. PPS-Systemen ungeeignet. Die Installation eines ERP-Systems wie SAP R/3 auf einem Demonstrations-Notebook (z. B. für Präsentationen beim Kunden) entspricht zwar einer nicht verteilten, d. h. auf einen einzelnen Rechner beschränkten Architektur, das System besteht jedoch aus der softwaretechnischen Sichtweise aus mehreren logisch getrennten Schichten und ist daher nicht monolithisch.

4.2 Zwei-Schichten-Architektur

Eine Zwei-Schichten-Architektur wird entweder durch die klassische Host-Kommunikation oder durch eine zweistufige Client-/Server-Architektur repräsentiert.

Bei der *Host-basierten Architektur (Fall a)* werden die Verarbeitungslogik und die Datenverwaltung auf einem zentralen Rechner (z. B. Mainframe) realisiert, der als Host fungiert. Die Schnittstelle zum Benutzer wird durch Terminals dargestellt.

Die *zweistufige Client-/Server-Architektur (Fall b)* sieht eine Trennung der Datenhaltung von der Verarbeitungslogik vor. Die Datenhaltung wird dabei von einem zentralen Datenbankserver übernommen, während die Verarbeitungslogik und die Präsentationsdienste durch so genannte Fat Clients übernommen werden. Im Gegensatz zu Fall a), der einen Einsatz „dummer" Terminals vorsieht, werden in Abhängigkeit von der zu unterstützenden Aufgabenstellung hohe Anforderungen an die Rechenleistung der Client-Rechner gestellt (z. B. wenn die Verarbeitungslogik die Berechnung von Sekundärbedarfen im Rahmen der Produktionsplanung vorsieht).

4.3 Drei-Schichten-Architektur

Bei einer Drei-Schichten-Architektur wird die Verarbeitungslogik des Anwendungssystems einer separaten Schicht zugeordnet. Die benötigten Dienste (z. B. spezielle Verfahren bzw. Algorithmen für die Produktionsplanung) werden auf der logischen Ebene durch einen oder mehrere *Applikationsserver* und auf der physischen Ebene durch zumeist mehrere separate Rechner bereitgestellt. Die redundante Verteilung auf mehrere leistungsstarke Rechner ermöglicht eine Verbesserung der in Fall 2b) oftmals unzureichender Performance. Eine Drei-Schichten-Architektur ist damit auch für den unternehmensweiten Einsatz mit einer großen Anzahl Daten, Transaktionen und Benutzern geeignet.

Thin Clients

Die Verlagerung der Verarbeitungslogik von den Clients hin zu separaten Applikationsservern führt zur Verwendung des Begriffes *Thin Client*. Thin Clients vereinen die Vorteile „dummer" Terminals (Fall 2b) wie einfache Administration und Wartung mit den Vorteilen von PC-Systemen wie z. B. der grafischen Benutzeroberfläche. Die Thin-Client-Architektur resultiert aus der Beobachtung, dass eingabeorientierte Anwendungen häufig den größten Teil der Arbeit repräsentieren [Ru02]. Der Rechner wartet demnach die meiste Zeit auf Benutzereingaben, um als Reaktion darauf Ausgaben auf dem Bildschirm anzuzeigen. Die Verwendung von Fat Clients führt folglich zu einer geringen Auslastung der Hardware-Ressourcen. Zudem erfordert die Übertragung der Benutzereingaben und der Bildschirmausgaben nur geringe Bandbreiten. Da moderne Server über ausreichend Rechen- und Speicherkapazität verfügen, ist der Gedanke nahe liegend, zu bereits von Host-basierten Architekturen bekannten Konzepten zurückzukehren und die Applikationen wieder vom Client auf den Server zu verlagern.

Server Based Computing

Der Thin-Client-Ansatz wird häufig in Verbindung mit dem Konzept des *Server Based Computing* betrachtet bzw. realisiert [Ru02]. Dabei stellen spezielle Server im Firmennetz nicht nur Datenbank- und Druckdienste, sondern auch alle nutzbaren Anwendungen bzw. Funktionen zur Verfügung. Dadurch lassen sich sowohl die Benutzer- als auch die

Applikationsverwaltung zentral vornehmen, so dass sich das bislang aufwändige Einspielen von Updates und Patches auf die Aktualisierung einiger weniger Server-Rechner reduziert.

Da auf den verwendeten Client-Rechnern nur das Betriebssystem und der Thin Client benötigt werden, kann die lokal verfügbare Festplattenkapazität stark reduziert werden oder entfallen (z. B. wenn direkt von CD gebootet wird). Da das Basisbetriebssystem keine weitreichenden Funktionen besitzt, ist der Benutzer auf die für ihn freigeschalteten Anwendungen beschränkt. Welche Anwendungen bereitzustellen sind, entscheidet das System anhand der Login-Informationen. Der Anwender findet somit stets seine persönliche Arbeitsumgebung vor, unabhängig von der Arbeitsstation, an der er sich befindet. Hinsichtlich der praktischen Umsetzung von Thin Clients und Server Based Computing existieren mehrere Philosophien wie z. B. Windows Based Terminals und Network Computer. *Windows Based Terminals (WBT)* sehen eine Verwendung der Terminal Services von Microsoft Windows oder der Produkte der Firma Citrix vor. WBT sind vollständig auf die Besonderheiten der Windows-Plattform abgestimmt, die Interaktion mit Java- oder Mainframe-Anwendungen kann jedoch evtl. mittels eines durch den Server zur Verfügung gestellten Web-Browsers oder einer Terminal-Emulation realisiert werden. Beim Einsatz von WBT werden alle Applikationen auf dem Server ausgeführt, und auch die Datenspeicherung erfolgt zentral auf dem Server. Lokal sind zu keiner Zeit Daten vorhanden, so dass der Client-Rechner keine Festplatte benötigt. Bei einem *Network Computer (NC)* handelt es sich um ein Thin-Client-System, das eine Java-basierte Technologie zur Realisierung eines Hybrid-Systems verwendet. Anwendungen können sowohl auf dem Server als auch auf dem Client als Java-Applets ausgeführt werden. NC verwenden keine Windows-spezifischen, sondern die im Internet üblichen Protokolle wie http [Ru02].

4.4 Web-basierte Mehr-Schichten-Architekturen

Die zunehmende Verbreitung von Web- bzw. Internet-Technologien lässt sich auch in Bezug auf Architekturen für ERP-Systeme beobachten. Bei einer Web-basierten ERP-Architektur wird der Thin Client durch einen Web-Browser repräsentiert, der eine einfache Bedienung von HTML-basierten Benutzeroberflächen ermöglicht. Geeignete Browser sind für alle relevanten Systemplattformen verfügbar, so dass bei der Entwicklung eines komplexen Anwendungssystems der Aufwand für die plattformspezifische Benutzeroberflächen-Programmierung weitgehend entfallen kann. Web-Technologien vereinfachen überdies den transparenten Zugriff auf Daten oder Funktionen von Systemen, die auf einem Mainframe-Rechner installiert sind. Zudem wird der Installations- und Wartungsaufwand für den Präsentationsclient reduziert [No00, Gr02].

Der im vorigen Kapitel erläuterte Aufbau von Drei-Schichten-Architekturen lässt sich in großen Teilen auf Web-basierte Architekturen übertragen. Durch die Verwendung eines *Web-Browsers* als Thin Client wird zusätzlich ein *Webserver* erforderlich, der die Bereitstellung von HTML-Dokumenten, Multimedia-Objekten oder Java-Applets übernimmt, die über das *Hypertext Transfer Protocol (HTTP)* vom Browser angefragt werden. Dar-

über hinaus werden weiterhin Applikations- und Datenbankserver benötigt [vA97, No00].

Das World Wide Web (WWW) wurde ursprünglich für die Abfrage statischer Dokumente konzipiert, die auf Webservern verteilt abgelegt sind. Um dialog- bzw. interaktionsorientierte Informationssysteme wie ein ERP-System zu realisieren, sind zusätzliche, neuere Internet-Technologien wie CGI-Abfragen und Java-Applets notwendig. Das *Common Gateway Interface (CGI)* ermöglicht die Weiterleitung von Anfragen aus dem Browser an beliebige Anwendungen. Der Webserver fungiert als Vermittler und gibt die Anfrage des Browsers über das CGI an den Applikationsserver weiter, der bei Bedarf mit dem Datenbankserver kommuniziert. Das Ergebnis wird über das CGI zurück an den Webserver übermittelt, dort in das HTML-Format übertragen, als *dynamisch erzeugte Webseite* an den Client gesendet und von diesem grafisch angezeigt [Gr02, No00, We02] (vgl. Abb. 7)

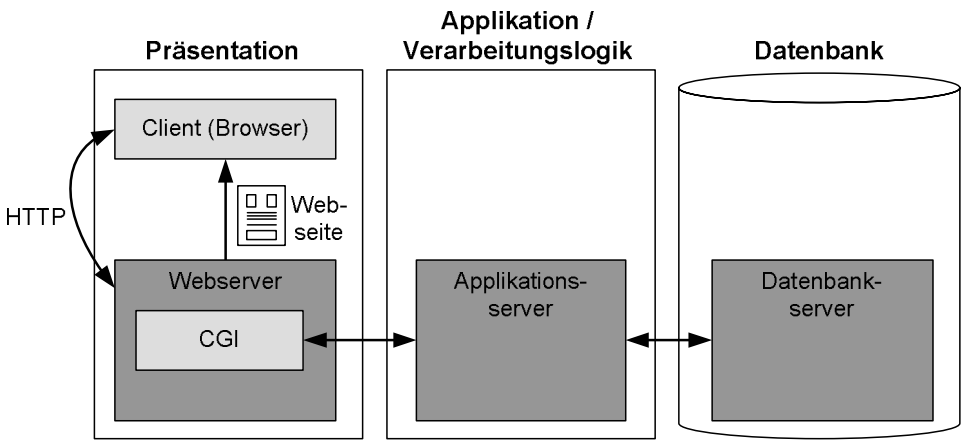

Abb. 7: CGI-basierter Zugriff auf Funktionen und Daten [Gr02]

Java-Applets, die als plattformunabhängige, eigenständige Programme durch den Browser vom Webserver angefordert und mittels der in den Browser integrierten Java Virtual Machine ausgeführt werden, können ohne Umweg über den Webserver direkt mit Applikations- oder Datenbankservern kommunizieren. Dabei kann bspw. das aus der Verknüpfung von WWW mit CORBA (Common Object Request Broker Architecture) entstandene *Internet Inter-ORB Protocol (IIOP)* verwendet werden. In gängigen Browsern ist ein Client-ORB (Object Request Broker) integriert, der mit Server-ORBs über das IIOP direkt kommunizieren kann (vgl. Abb. 8).

Neben IIOP existieren weitere, alternative Protokolle, die für den Nachrichtenaustausch zwischen Clients und Servern verschiedener Architekturschichten verwendet werden können. Dazu gehören bspw. SOAP *(Simple Object Access Protocol)*, das auf HTTP und auf XML (Extensible Markup Language) basiert und keinen ORB benötigt, und *DCOM (Distributet Component Object Model, zukünftig COM+)*, das von Microsoft entwickelt wurde und auf der Windows-Plattform basiert. Die Dienste, die ein Client über SOAP

oder ein alternatives Protokoll in Anspruch nimmt, werden oftmals als *Web Services* bezeichnet [An00, WW02, St02, No00].

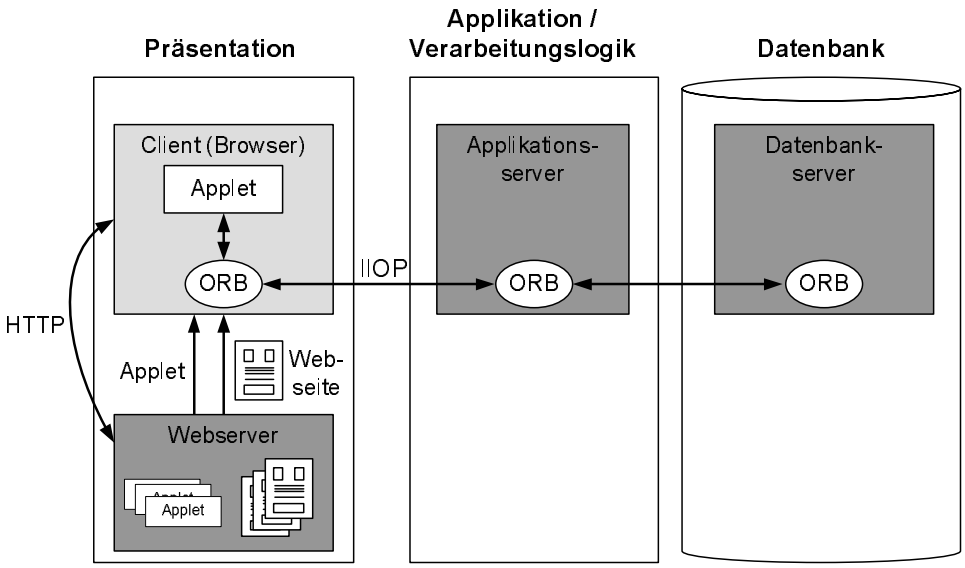

Abb. 8: Java- und CORBA-basierter Zugriff auf Funktionen und Daten [Gr02]

4.5 Komponentenorientierung

Die Vorteile Web-basierter Architekturen sind offensichtlich, wenn Plattformunabhängigkeit, Skalierbarkeit und hohe Integrationsfähigkeit gefordert werden [SKM00]. Kann eine Windows-basierte Systemlandschaft vorausgesetzt werden, ermöglichen WBT-Lösungen wie Citrix MetaFrame eine schnell und einfach zu realisierende sowie auf die Windows-Plattform abgestimmte Thin-Client-Architektur. Als zukünftige Entwicklung wird häufig eine zunehmende Komponentenorientierung genannt, die durch die voranschreitende Standardisierung von Kommunikationsprotokollen wie SOAP und von Dienste-Verzeichnissen (z. B. UDDI - Universal Description, Discovery and Integration) eine unternehmensindividuelle Konstruktion integrierter Informationssystem-Architekturen durch die Verknüpfung von Komponenten bzw. Web-Services ermöglicht („Best-of-breed“-Ansatz).

In Bezug auf komplexe ERP-Systeme ist diese Entwicklung jedoch zweifelhaft, denn vor dem Hintergrund der strategischen Bedeutung einer ERP-Auswahl und des Bestrebens von Industrieanwendern, Anbieter wie z. B. SAP zu favorisieren, bei denen davon ausgegangen werden kann, dass sie langfristig am Markt existent sein werden, erscheint die Verbreitung solcher ERP-Architekturen unrealistisch, die aus vielen einzelnen Komponenten unterschiedlicher Anbieter bestehen. Zudem würde eine umfassende Realisierung der Komponentenorientierung zu einer großen Heterogenität der benötigten Qualifikati-

onen von IT-Unternehmensberatern und internen Mitarbeitern führen. Außerdem bleibt das Problem einer vernünftigen Datenintegration (auf semantischer und struktureller Ebene) beim Einsatz unterschiedlich strukturierter Fachkomponenten ungelöst. Potenziale der Komponentenorientierung in Verbindung mit Werkzeugen für Enterprise Application Integration (EAI) und mit Internet-basierten Kommunikationsstandards sind daher eher in der einfacheren Ergänzung bestehender ERP-Architekturen um zusätzlich benötigte Funktionalität bzw. in der einfacheren Integration von ERP-Systemen mit anderen Anwendungssystemen der bestehenden DV-Landschaft zu sehen.

Anwender fordern darüber hinaus zunehmend standardisierte Mechanismen für die einfache Kommunikation mit Systemen ihrer Marktpartner im Rahmen des Supply Chain Managements, wobei nach wie vor Probleme wie unzureichende Kommunikationsstandards, Performance-Probleme und fehlende Plausibilitätskontrollen für die Übertragung von Informationen bestehen. Beispielsweise lassen sich XML-Daten auf syntaktische Korrektheit prüfen, die semantische Fehlerfreiheit kann jedoch zumeist nicht systemgestützt überprüft werden.

5 Zusammenfassung

Informationsmodelle spielen als Entscheidungsobjekte für Informationsmanager eine zentrale Rolle. Sie werden zur Spezifizierung und Konkretisierung betriebswirtschaftlicher und DV-technischer Sachverhalte verwendet. Sie dienen somit u. a. für die Auswahl oder die Entwicklung von Anwendungssystemen, aber auch zur Schaffung einer einheitlichen Diskussionsgrundlage und Abgrenzung des Gegenstandsbereichs. Informationssystem-Architekturen als eine Art von Informationsmodellen nehmen dabei eine besondere Rolle ein. Durch die Darstellung auf einem hohen Abstraktionsgrades ermöglichen sie eine Strukturierung über mehrere Informationsmodelle hinweg. Informationsmanager strukturieren den Domänenbereich in fachkonzeptionellen Architekturen wie dem Handels-H-Ordnungsrahmen. Entscheidungen über fach- und DV-konzeptionelle Ausprägungen werden bei der Gestaltung einer Architektur für Workflow-basierte PPS-Systeme gefällt, während Entscheidungen über Mehr-Schichten-Architekturen der DV-Konzept-Ebene zuzuordnen und domänenneutral sind. Die wesentliche Gestalt des Gesamtinformationssystems einer Unternehmung sollte als Leitlinie in einer oder mehreren Architekturen festgelegt sein.

6 Literaturverzeichnis

[Ad98] Adam, D.: Produktions-Management. 9. Aufl., Wiesbaden, 1998.

[An00] Anastasopoulos, M. et al.: Webanwendungen: Überblick Stand der Technik. Technical Report. Project "ApplicationToWeb", 2000. http://app2web.fzi.de/themen/ap1/state_of_the_art.pdf (2003-04-25).

[Be01] Becker, J. et al.: Konstruktion von Methodiken: Vorschläge für eine begriffliche Grundlegung und domänenspezifische Anwendungsbeispiele. In (Becker, J. et al., Hrsg.): Arbeitsberichte des Instituts für Wirtschaftsinformatik. Nr. 77. Münster 2001.

[BHN02] Becker, J.; Hansmann, H.; Neumann, S.: Workflow-basierte Produktionsplanung und -steuerung. Ein Architekturmodell für die Koordination von Prozessen der industriellen Auftragsabwicklung. In (Loos, P.; Gronau, N., Hrsg.): E-Business - Integration industrieller ERP-Architekturen. Göttingen, 2002, S. 17-29.

[BM03] Becker, J.; Meise, V.: Strategie und Ordnungsrahmen. In (Becker, J.; Kugeler, M.; Rosemann, M., Hrsg.): Prozessmanagement. Ein Leitfaden zur prozessorientierten Organisationsgestaltung. 4. Aufl., Berlin et al., 2003, S. 107-157.

[BS03] Becker, J.; Schütte, R.: Handelsinformationssysteme. 2. Aufl., Landsberg/Lech, 2003.

[BV96] Becker, J.; Vossen, G.: Geschäftsprozeßmodellierung und Workflow-Management: Eine Einführung. In: (Vossen, G.; Becker, J., Hrsg.): Geschäftsprozeßmodellierung und Workflow-Management. Bonn et al., 1996, S. 17-26.

[FFG97] Fandel, G.; François, P.; Gubitz, K.-M.: PPS- und integrierte betriebliche Softwaresysteme. Grundlagen, Methoden, Marktanalyse. 2. Aufl., Berlin et al., 1997.

[Gr02] Grabowski, H.: Informationssysteme in der Produktions- und Konstruktionstechnik (IDPK). Vorlesungsskript, Universität Karlsruhe. Karlsruhe, 2002. http://www-rpk.mach.uni-karlsruhe.de/vorlesungen/IdPK.htm (2003-04-24).

[GS95] Galler, J.; Scheer, A.-W.: Workflow-Projekte: Vom Geschäftsprozeßmodell zur unternehmensspezifischen Workflow-Anwendung. In: IM - Information Management, 10 (1995) 1, S. 20-28.

[Ha03] Hansmann, H.: Architekturen Workflow-gestützter PPS-Systeme – Referenzmodelle für die Koordination von Prozessen der industriellen Auftragsabwicklung von Einzel- und Kleinserienfertigern. Dissertation, Universität Münster, 2003.

[HNB01] Hansmann, H.; Neumann, S.; Becker, J.: Workflow in der industriellen Auftragsabwicklung. In: Zeitschrift für Unternehmensentwicklung und Industrial Engineering (FB/IE), 50 (2001) 4, S. 153-157.

[HR89] Heinrich, L. J.; Roithmayr, F.: Wirtschaftsinformatik-Lexikon. 3. Aufl., München, Wien, 1989.

[Ko02] Kossmann, D.: Web Informationssysteme. Folien zur Vorlesung. Technische Universität München, München, 2002. http://www3.informatik.tu-muenchen.de/lehre/WS2002/XML-kossmann/kap0.pdf (2003-04-25).

[Kr90] Krcmar, H.: Bedeutung und Ziele von Informationssystem- Architekturen. In: Wirtschaftsinformatik, 32 (1990) 5, S. 395-402.

[Ku99] Kurbel, K.: Produktionsplanung und -steuerung. Methodische Grundlagen von PPS-Systemen und Erweiterungen. 4. Aufl., München et al., 1999.

[LBL02] Luczak, H.; Becker, J.; Lassen, S.: Workflow-Management in der Produktionsplanung und -steuerung. Effizienz in der Auftragsabwicklung erhöhen. In: wt Werkstattstechnik online 92 (2002) 5, S. 256 - 258.

[LE99] Luczak, H.; Eversheim, W.: Einführung. In (Luczak, W.; Eversheim, H., Hrsg.): Produktionsplanung und -steuerung. Grundlagen, Gestaltung und Konzepte. 2. Aufl., Berlin, 1999, S. 3-5.

[Lo98] Loos, P.: Workflow-Tauglichkeit von PPS-Funktionen - Typologien und Einsatzszenarien. In (von Uthmann, C. et al., Hrsg.): Proceedings zum Workshop "PPS meets Workflow". Gelsenkirchen, 1998, S. 24-36.

[Me01] Meise, V.: Ordnungsrahmen zur prozessorientierten Organisationsgestaltung. Modelle für das Management komplexer Reorganisationsprojekte. Hamburg, 2001.

[MH03] zur Mühlen, M.; Hansmann, H.: Workflowmanagement. In (Becker, J.; Kugeler, M.; Rosemann, M., Hrsg.): Prozessmanagement. Ein Leitfaden zur prozessorientierten Organisationsgestaltung. 4. Aufl., Berlin et al., 2003, S. 385-421.

[Ne03] Neumann, S.: Workflow-Anwendungen in technischen Dienstleistungen. Eine Referenz-Architektur für die Koordination von Prozessen im Gebäude- und Anlagenmanagement. Dissertation, Universität Münster. Münster, 2003.

[No00] Nocak J. et al.: Architekturen für Network Computing. In: Wirtschaftsinformatik, 42 (2000) 1, S. 5-14.
[Ru02] Rubner, S.: Thin Clients - die schlanke Alternative. White Paper 2002. http://www.edv-knowhow.net/Dokumente/Netzwerk/Thin_clients.pdf (2003-04-25).
[Sa95] Sager, W.: Von zentraler IV zur Client/Server-Architektur. In: Wirtschaftsinformatik, 37 (1995) 3, S. 294-302.
[Sc01] Scheer, A.-W.: ARIS - Modellierungsmethoden, Metamodelle, Anwendungen. 4. Aufl., Berlin et al., 2001.
[Sc97] Scheer, A.-W.: Wirtschaftsinformatik. Referenzmodelle für industrielle Geschäftsprozesse. 7. Aufl., Berlin et al., 1997.
[Sc93] Schmitt, H.-J.: Client-Server Architekturen. Architekturmodelle für eine neue informationstechnische Infrastruktur. Frankfurt/Main 1993.
[SKM00] Sinz, E. J.; Knobloch, B.; Mantel, S.: Web-Application-Server. In: Wirtschaftsinformatik, 42 (2000) 6, S. 550-552.
[SNZ95] Scheer, A.-W.; Nüttgens, M.; Zimmermann, V.: Rahmenkonzept für ein integriertes Geschäftsprozeßmanagement. In: Wirtschaftsinformatik, 37 (1995) 5, S. 426-434.
[St02] Stiemerling, O.: Web-Services als Basis für evolvierbare Softwaresysteme. In: Wirtschaftsinformatik, 44 (2002) 5, S. 435 - 445.
[Te99] Teubner, R. A.: Organisations- und Informationssystemgestaltung. Theoretische Grundlagen und integrierte Methoden. Wiesbaden, 1999.
[To02] Tomczyk, P.: Mehrschichtenarchitekturen und Komponenten-Frameworks. Folien zur Vorlesung Informationsintegration und Web-Portale. Universität Karlsruhe, Karlsruhe, 2002. http://hestia.fzi.de/klick-and-bau/WS0203/02.ppt (2003-04-25).
[vA97] von Arb, R.: Vorgehensweisen und Erfahrungen bei der Einführung von Enterprise-Management-Systemen dargestellt am Beispiel von SAP R/3. Dissertation, Universität Bern. Bern, 1997.
[We02] Weitz, W.: Basisarchitekturen Web-basierter Informationssysteme. In: Wirtschaftsinformatik, 44 (2002) 3, S. 207-216.
[WW02] World Wide Web Consortium (W3C): Web Services Architecture. W3C Working Draft (14. November 2002). http://www.w3.org/TR/2002/WD-ws-arch-20021114/ (2003-04-25).
[Zä98] Zäpfel, G.: Grundlagen und Möglichkeiten der Gestaltung dezentraler PPS-Systeme. In (Corsten, H.; Gössinger, R., Hrsg.): Dezentrale Produktionsplanungs- und -steuerungs-Systeme: eine Einführung in zehn Lektionen. Stuttgart et al., 1998, S. 13-53.