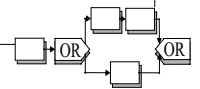


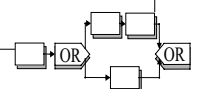
Kapitel 5: Flexibles Workflow-Management

- Problembeschreibung
- Klassifikation von Ansätzen
- Ausgewählte Systeme
- Zusammenfassung und Diskussion

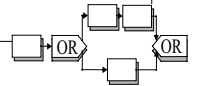
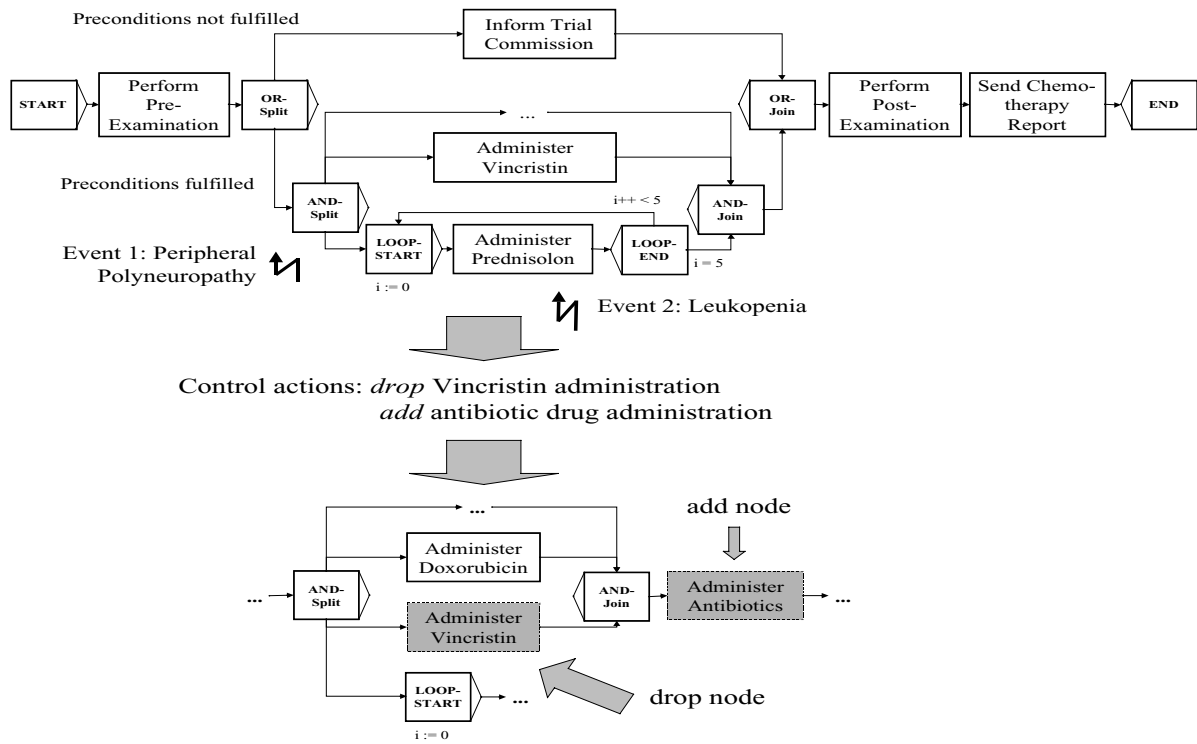


Ausgangssproblematik

- Die zum Zeitpunkt einer Workflow-Definition getroffenen Annahmen müssen zur Ausführungszeit nicht mehr gelten
- Dynamischen Veränderungen unterworfen sind
 - Ressourcensituation (Personal, Rechner, Geräte, ...)
 - Zeitliche Vorgaben (z.B. Veränderung bzgl. Deadlines)
 - Menge der notwendigen und sinnvollen Aktivitäten (Kontrollfluß)
 - Datenfluß
- Daher: Workflow-System muß flexibel auf Veränderungen reagieren können
- Teilaspekte
 - Erkennung der veränderten Situation
 - Dynamische Anpassung (manuell oder automatisch)
 - Verifikation der Anpassung
- Derzeit kaum Unterstützung durch kommerzielle Systeme



Flexibles Workflow-Management: Motivationsbeispiel



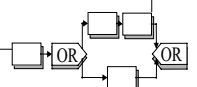
Klassifikation von Ansätzen (1)

• Vollständige Vormodellierung

- Mögliche Abweichungen bzgl. der Annahmen werden vormodelliert
- Konditionale Elemente in Workflow-Definition
- Vorteil: Keine Adaptation zur Laufzeit nötig
- Nachteile:
 - Mögliche Abweichungen (und ihr relativer Zeitpunkt) müssen *bekannt* sein
 - Unübersichtliche Workflows, Vermischung von Normal- und Ausnahmefall

• Late Modeling/Late Binding

- Bestimmte Aspekte werden zur Definitionszeit offen gelassen bzw. unterspezifiziert
- Syntaktisch durch "Placeholder"-Elemente (z.B. abstrakte Knoten)
- Zur Ausführungszeit Konkretisierung des offenen Aspektes
- Üblich für Ressourcenzuweisung
- Analogie zu Programmiersprachen: Late Binding von Methoden
- Vorteil: Relative einfache Handhabung zur Laufzeit, übersichtlicher als Vormodellierung
- Nachteil: Offene Aspekte (und ihr relativer Zeitpunkt) müssen *bekannt* sein



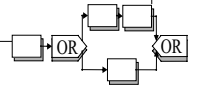
Klassifikation von Ansätzen (2)

• Ad-hoc-Adaptation

- Ausspezifizierter Workflow
- Bei Bedarf Umbau zur Laufzeit
- Vorteile:
 - Weniger Annahmen über Art und relativen Zeitpunkt einer veränderten Situation
 - Größere Flexibilität
 - Klare Trennung von Normal- und Ausnahmefall
- Nachteile:
 - Tiefgreifender Eingriff in Workflow-Instanz
 - Korrektheit der veränderten Instanz muß gewährleistet werden

• Schema-Evolution

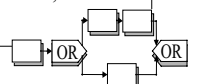
- Veränderung von Workflow-Definitionen
- Nötig aufgrund veränderten Wissens
- Problematik: Behandlung von Instanzen, deren zugrundeliegende Definition geändert wurde



Late Modeling: Beispiel MOVE ²

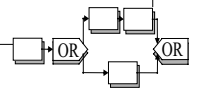
- Workflow-Modellierung basiert auf FunSoft-Netzen
- Für jede Workflow-Instanz wird eigene Netzkopie verwaltet
- Sub-Netze mit nicht oder nur teilweise planbarer Ablaufstruktur werden zur Laufzeit dynamisch definiert (*Late Modeling*)
- Hierarchische Transitionen, für die zur Laufzeit ein Sub-Netz dynamisch modellierbar sein soll, müssen explizit gekennzeichnet werden
- Spätes Modellieren muß vor Instanziierung der Transition bzw. ihres Sub-Netzes abgeschlossen sein

² Hagemeyer, J. et al.: *Flexibilität bei Workflow-Management-Systemen*. Likowski, R. et al. (Hrsg.): Proc. Software-Ergonomie'97, Dresden, März 1997, S. 179–190.



MOVE: Beispiel für späte Netz-Modellierung

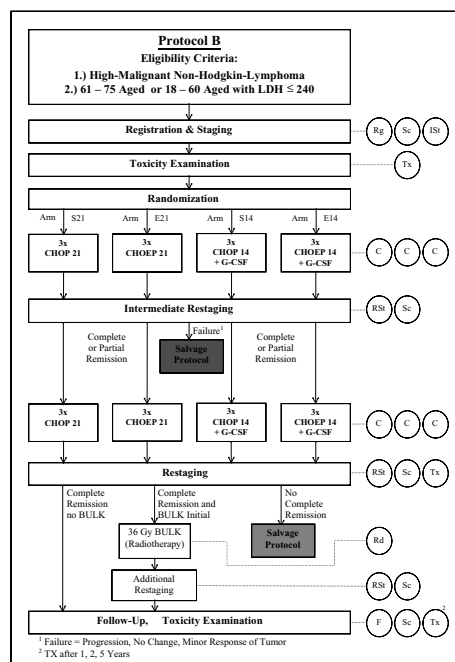
- Sub-Netz der Transition B kann (manuell oder automatisch) editiert werden, wenn die gekennzeichnete Stelle s mit einer Marke belegt wird
- Die Transition C kann parallel zu diesem Editiervorgang ausgeführt werden



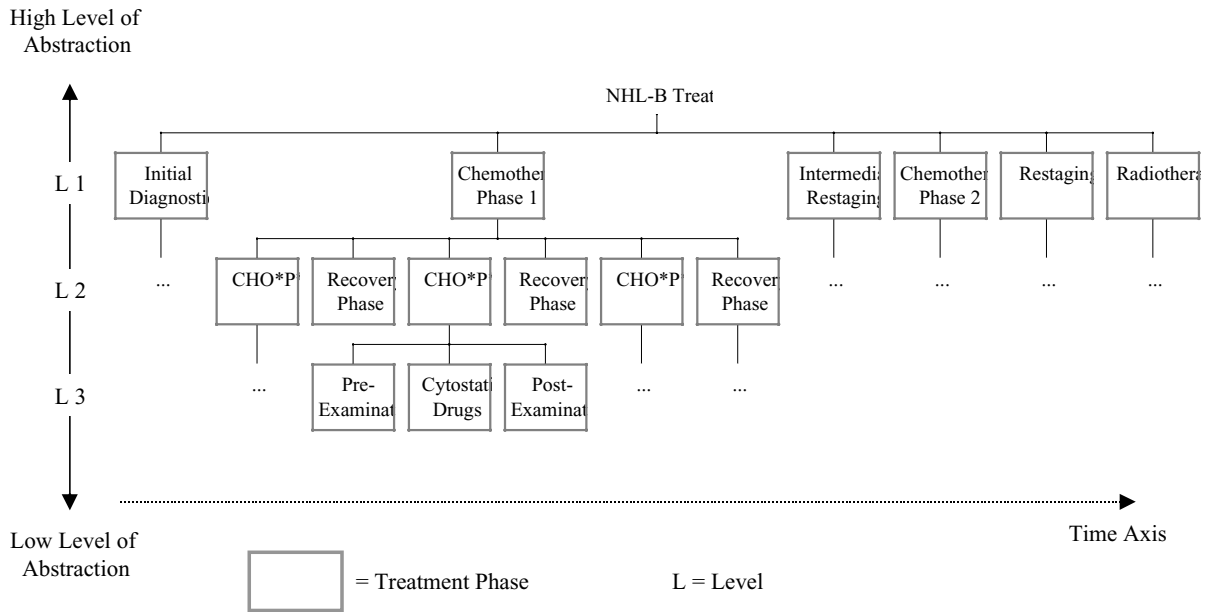
(C) Prof. E. Rahm, R. Müller

7

Late Binding: Beispiel HEMATOWORK (Uni Leipzig)

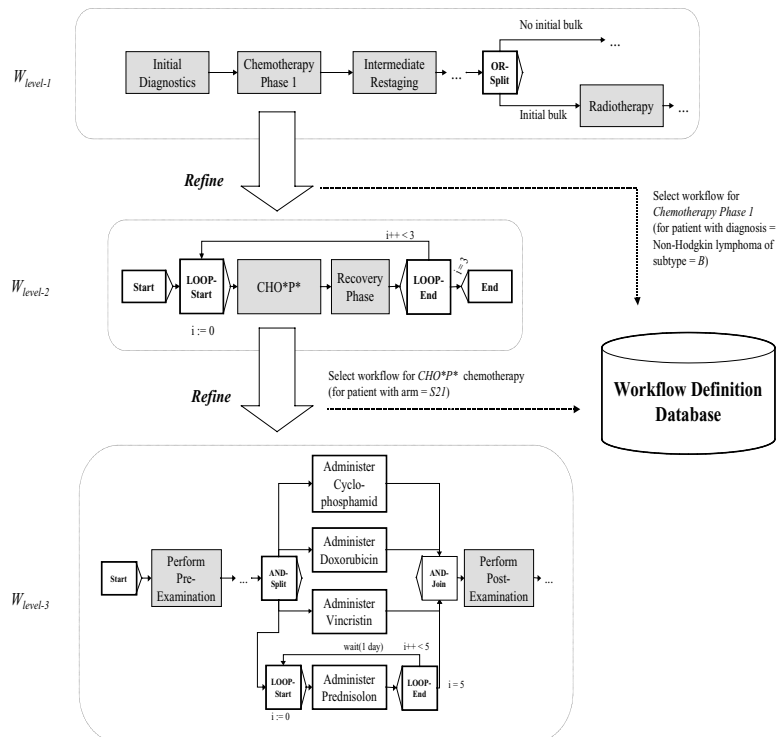


Beispiel HEMATOWORK: Hierarchische Protokolle



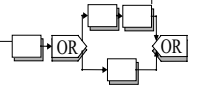
Late Binding in HEMATOWORK

- Placeholder-Nodes repräsentieren Subworkflows
- Zur Laufzeit Auswahl eines geeigneten Subworkflows
- Integration in Gesamt-Workflow
- Synchrone Ausführung des Subworkflows



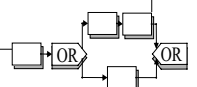
Late Modeling/Late Binding: Zusammenfassung

- Sinnvoll einsetzbar in vielen Anwendungen
- Unterstützt hierarchischen Aufbau von Workflows
 - Hierarchische Verfeinerung
 - Nicht zu verwechseln mit *statischen* Subworkflows
- Einfach zu realisieren, da keine tiefgreifende Strukturänderung des Workflows
- Modifizierbarkeit des Workflows auf vorab definierte Bereiche eingeschränkt
- Nicht planbare Anteile müssen zur Modellierungszeit bekannt sein
- Bei manuellem Late Modeling: Endanwender muß Definitionssprache beherrschen



Ad-hoc-Adaptation

- Auspezifizierter Workflow wird ausgeführt
- Bei Bedarf strukturelle Veränderung des Kontroll- und Datenflusses
 - Benutzerinduziert
 - Ereignisorientiert
 - Manuell oder automatisch
- Erfordernisse
 - Geeignete Repräsentation von Workflow-Instanzen (Kopie der Definition)
 - Graphtechnische Operatoren für strukturelle Änderungen
 - Konsistenzkriterien
 - Verifikation
 - Möglichst hohes Abstraktionslevel um Benutzerinteraktion zu unterstützen

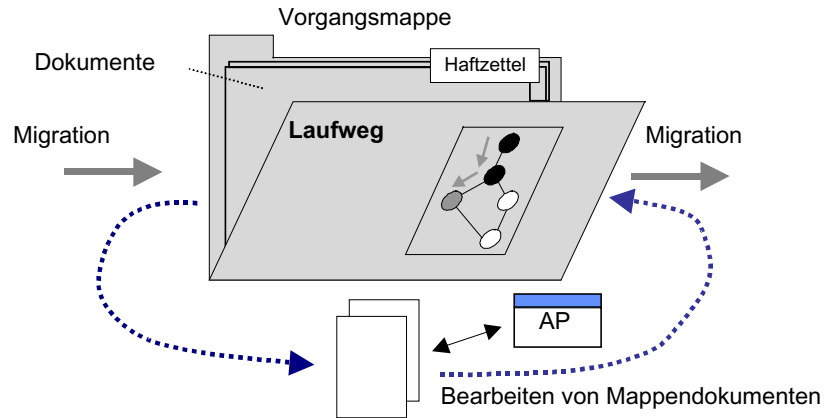


Ad-hoc-Adaptation bei kommerziellen Produkten

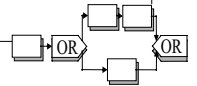
• Kaum Unterstützung bei den meisten Produkten

• Beispiel ProMInanD²

- Dokumentenorientiertes WfMS
- Modell der elektronischen Umlaufmappe
- Unterstützung einfacher Änderungen



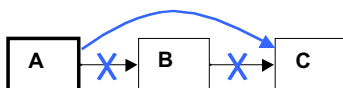
2 Karbe, B.: *Flexible Vorgangsteuerung mit ProMInanD*. In: Hasenkamp, U. et al. (Hrsg.): *CSCW – Computer Supported Cooperative Work*. Addison-Wesley, 1994, S. 117–133



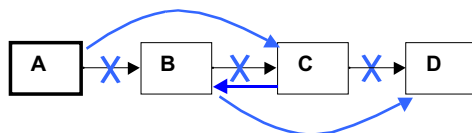
Ad-hoc-Adaptation in ProMInanD

• Mögliche Operationen

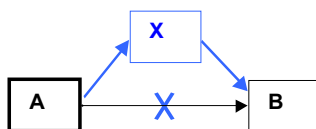
a) Überspringen



b) Verschieben



c) Einfügen

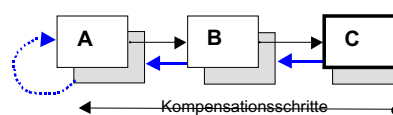


d) Zurückspringen



Vorgang zurückholen, Bearbeitung ablehnen

e) Zurücksetzen

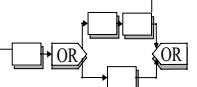


■ Aktuell beendeter Schritt

□ Aktivierter, aber noch nicht gestarteter Schritt

• Limitationen

- Lediglich Unterstützung weniger, einfach zu realisierender Operationen (z.B. Einfügen eines neuen Schrittes direkt nach Beendigung einer Aktivität und vor Weiterleiten des Vorgangs an nächste Arbeitsstation)
- Keine formale Analyse von Datenflüssen



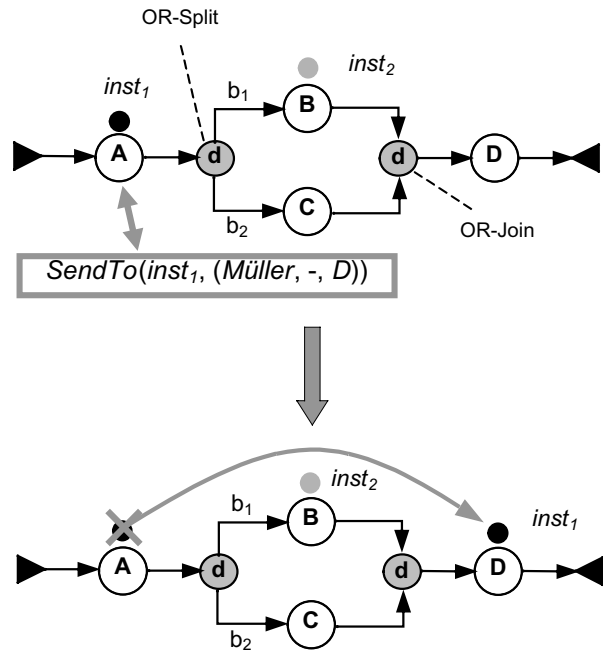
Ad-hoc-Adaptation in Petri-netz-basierten WfMS

Adaptation durch dynamische Änderung von Netzmarkierungen

- Realisierung von einfachen Ad-hoc-Eingriffen in Workflow-Kontrolle (z.B. Vorwärts- / Rückwärtssprung) durch dynamische Änderung von Netzmarkierungen

Beispiel: Chautauqua²

- Basiert auf höherem Petri-Netz-Formalismus (z.B. gefärbte Tokens, spezielle Knoten für UND-/ODER-Ausführung)
- Ad-hoc-Eingriffe in Workflow-Kontrolle durch Änderung der Token-Lokalisation
- Beispiel: Überspringen von Aktivitäten durch Verschieben von Marken
- Kritik: Geringes Abstraktionsniveau; Benutzer muß Tokens "per Hand" verschieben
- Inkonsistente Zustände (z.B. Rücksprünge in tote Zweige) und Verklemmungen möglich



² Ellis, C.A.; Maltzahn, C.: *The Chautauqua Workflow System*. Proc. 30th Hawaii Int'l Conf. on System Sciences, Maui, Hawaii, Januar 1997

