Flexibilität durch Selbstorganisation in Workflow-Management-Systemen

Stefan Schreyjak

Universität Stuttgart
Institut für Parallele und
Verteilte Höchstleistungsrechner

Gliederung des Vortrags:

1. Begriff Flexibilität im WFM
2. Problemstellung bei Selbstorg.
3. Mein Lösungsansatz:
   - Componentware in Aktivität
   - Prozeßkontext

Stefan.Schreyjak@informatik.uni-stuttgart.de
http://www.informatik.uni-stuttgart.de/ipvr/as/personen/schreyjak.html

Flexibilität und Selbstorganisation

Flexibilität:

Fähigkeit, sich an neue oder sich verändernde Anforderungen anpassen zu können.

- **Abänderbarkeit:**
  Vorhandenes ändern
- **Erweiterbarkeit:**
  Um etwas neues oder ähnliches erweitern

Selbstorganisation:

Organisationskonzept, Koordination der Arbeit durch autonom kooperierende Benutzer

Feststellung: Selbstorganisation erfordert Flexibilität

Schichtenmodell

Wie beurteilt man den Grad der Flexibilität?

1. Nehme Daten, die System repräsentieren
2. Teile Daten in Schichten ein gemäß:

Information Resource Dictionary Systems (IRDS) [ISO-Standard 1990]

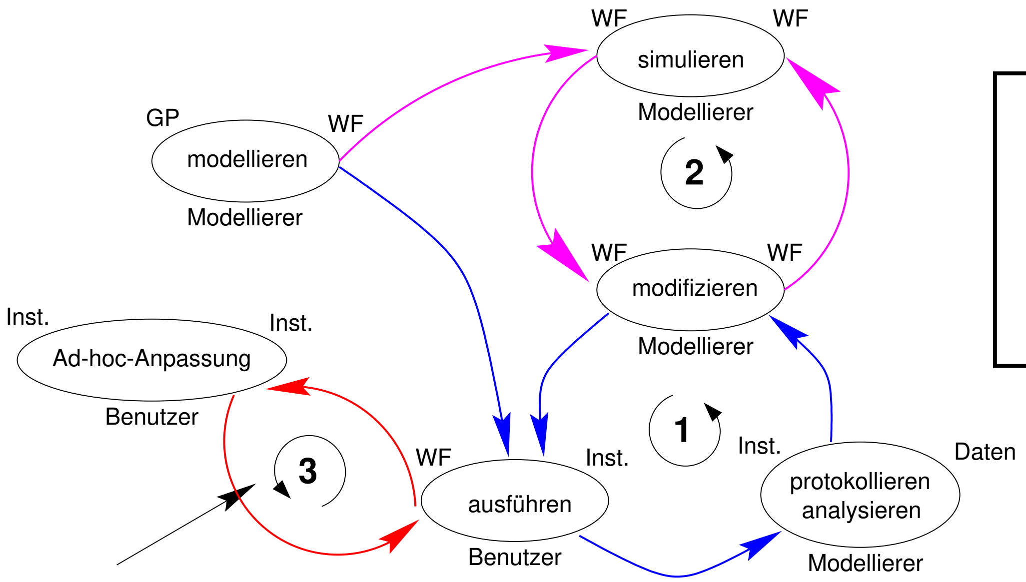
	DBMS	WFMS	Programme
3	Schemabeschreibung	Workflow-Sprache	Programmiersprache
2	Schema	Workflow-Schema	Quelltext
1	Anwendungsdaten	Workflow-Instanz	Programminstanz

Schichten sind Meta-Ebenen der Systembeschreibung.

Flexibilität ist größer, je mehr Daten pro Schicht änderbar sind

Einsatzprozeß

Flexibilität zeigt sich auch im Einsatzprozeß eines Workflows:



Nutzen der Flexibilität durch Zyklen!

Wenn Workflow *a priori* nicht präzise beschreibbar ist, kann eine Anpassung der Instanz zur Laufzeit notwendig werden.

Problemstellung

Vorhandene Zyklen sind meist ausreichend.

Was aber bei Selbstorganisation?

z.B. die Modifikation der Aktivitätsimplementierungen!

Forderung 1:

Benutzer soll Anwendungsprogramme erstellen und ändern/ anpassen können! (zumindest manche)

Keine Teamunter-stützung!

Anpassungen sind bisher entweder lokal oder global.

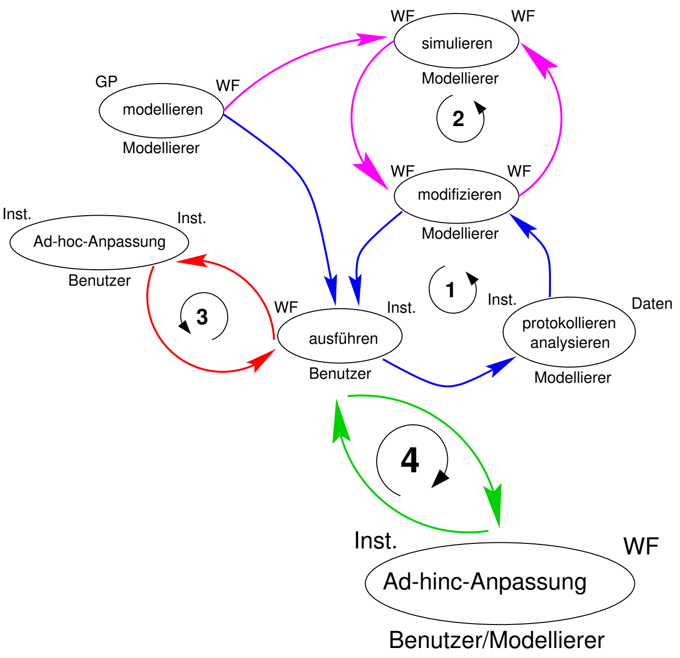
Forderung 2: 3 1/2

Unterschiedliche Reichweiten der Anpassung werden notwendig!

Nur Programminstanz anpassen lohnt sich nicht.
Einschränkung der Anpassung auf bestimmte Gruppen, Umstände, Randbedingungen

Lösungsansatz

Neuen Zyklus einführen: "Ad-hinc" (von jetzt an)



Zyklus 1/2:
Anpassung des WF-Schemas

Zykus 3:
Anpassung einer WF-Instanz

Zyklus 4:
Anpassen von Anwenderprogrammen durch Selbstorganisation

Inkrementelles Entwickeln wird möglich!

Komponentenorientiertes System

Wie soll Benutzer Anwendungsprogramme erstellen oder ändern ?

Durch ein komponentenorientiertes System!

- Vorgefertigte Komponenten als Bausteine
- Zusammenstecken wie bei einem Verbunddokument
- Verschiedene Aspekte modellieren z.B. Struktur, Layout, Verhalten

→ Ähnlichkeit zum Workflow-Management-Ansatz!

Eine Art "Mini-Workflow-System" innerhalb einer Aktivität!

Prozeßkontext

Wie differenzierte Reichweiten der Anpassungen erzeugen?

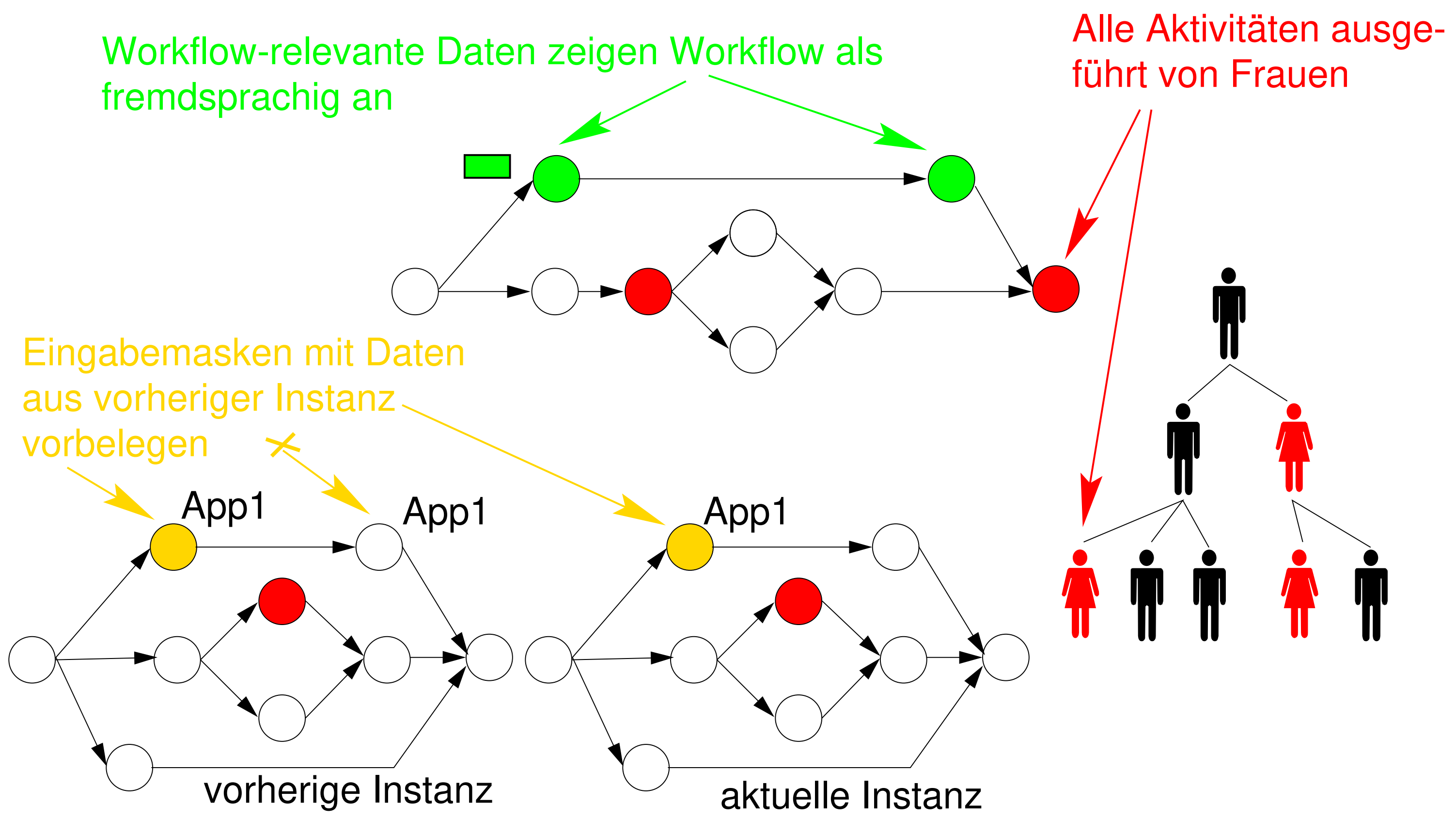
Ziel: Eine Anpassung nicht nur lokal oder global durchführen,
sondern auch auf bestimmbare Gruppen von Aktivitäten

Kopplung zwischen KOS und WFMS nötig:

- Spezifikation des Kontextes als Gruppe von Aktivitäten im WFS
- Anstoßen und Durchführen der Anpassung beim Aufruf der KOS
- Zuordnen einer Anpassung zu einem Kontext

Workflow-System steuert Anpassungen!

Beispiele für Prozeßkontexte



Universität Stuttgart
Software-Labor
Projekt 1
Workflow Management

Zusammenfassung

- Flexibilität
  - änderbare Meta-Schichten
  - Modifikationszyklen
- Selbstorganisation erfordert ein flexibles Workflow-System
- Komponentenorientiertes System zum Erstellen und Anpassen von Anwendungsprogrammen in Aktivitäten durch Benutzer.
- dafür Ad-hinc-Zyklus einführen
- Anpassungen mit unterschiedlicher Reichweite durch
- Prozeßkontext
  - repräsentiert Reichweite
  - definiert durch Gruppe von Aktivitäten

Stefan Schreyjak *Flexibilität durch Selbstorganisation in WFMS* Workshop: Flexibilität und Kooperation in WFMS