A Declarative Framework for Enterprise Information Systems

Qualification Report

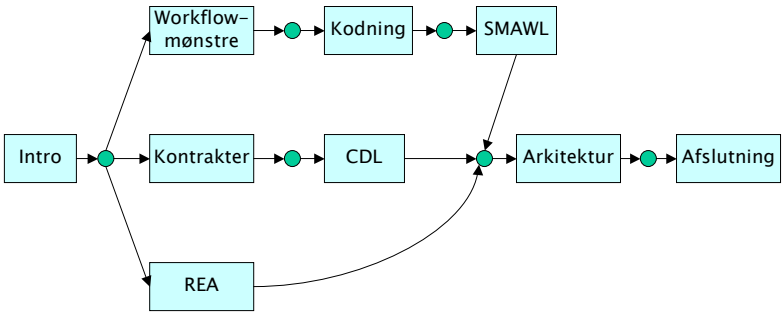
Christian Stefansen
cstef@diku.dk

NEXT

August 2005

Dagens workflow

Agenda



2

Problem

Agenda

At skrive, automatisere og overvåge forretningsgange i virksomhedssystemer i en service-orienteret arkitektur.

Nuværende systemer er ikke process-orienterede:

1. ad hoc-integration mellem moduler,
2. deler ikke processer med omverdenen,
3. understøtter ikke mobilitet.

3

Opgave

Agenda

Design en process-orienteret programmeringsplatform

- **Processer** og regler
- **Datamodel**
- **Rapportering** og overvågning

4

Processer

Agenda

- Concurrency-modeller
  - Processkalkuler
  - Petrinet
- Workflows
- BPEL – Business Process Execution Language
- YAWL – Yet Another Workflow Language

5

Hvorfor formelle workflows?

Agenda

- Brugere
  - kan let følge etableret praksis
  - ved hvilke opgaver der kan løses nu/senere
  - får hjælp til delegering af opgaver
  - behøver mindre viden om hele workflowet
- Designere/planlæggere
  - kan lettere kortlægge og ændre processer
  - kan indføre struktur *ad hoc*
  - kan foretage formel analyse
  - kan delvist automatisere outsourcing og lign.
- Controllere
  - får finere omkostningsregistrering (à la ABC)
  - kan nemmere foretage præstationsanalyse

6

Workflowmønstre

Agenda

- [Aalst 02-05]:
  - 20 mønstre for kontrolvej (parallel, sekvens, valg, gentagelse etc.)
  - 39 mønstre for datahåndtering/-vej (virkefelt, parameteroverførsel etc.)
  - 43 mønstre for delegering (implicit, pr. kø, pr. rolle, sag, revidering)
- Fokus her er kontrolvejsmønstre

7

De 20 kontrolvejsmønstre

Agenda

Basic Control Patterns

1 Sequence
2 Parallel Split
3 Synchronization
4 Exclusive Choice
5 Simple Merge

Advanced Branching and Synchronization Patterns

6 Multiple Choice
7 Synchronizing Merge
8 Multiple Merge
8a N-out-of-M Merge *(new)*
9 Discriminator
9a N-out-of-M Join

Patterns Involving Multiple Instances

12 MI without synchronization
13 MI with a priori known design time knowledge
14 MI with a priori known runtime knowledge
15 MI with no a priori runtime knowledge

Structural Patterns

10 Arbitrary Cycles
11 Implicit Termination

Cancellation Patterns

19 Cancel Activity
20 Cancel Case

State-Based Patterns

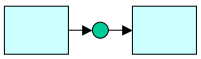
16 Deferred Choice
16a Deferred Multiple Choice *(new)*
17 Interleaved Parallel Routing
18 Milestone

8

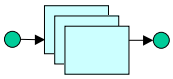
Nogle mønstereksempler

Agenda

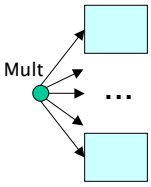
Pattern 1: Sequence
Execute activities in sequence.



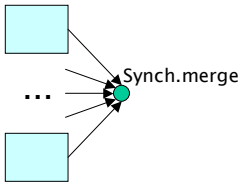
Pattern 13: Multiple Instances [...]
Generate many instances of one activity [...] with synchronization.



Pattern 6: Multiple Choice
Choose several execution paths from many alternatives.



Pattern 7: Synchronizing Merge
Merge many execution paths.
Synchronize if many paths are taken.
[...] Merge if only one [...] path is taken.

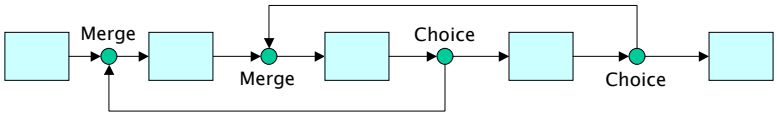


9

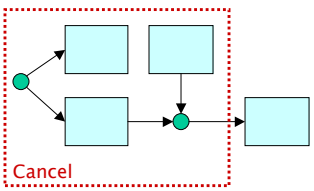
Flere mønstereksempler

Agenda

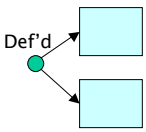
Pattern 10: Arbitrary Cycles
Execute workflow graph without any structural restriction on loops.



Pattern 20: Cancel Case
Cancel (disable) the process)



Pattern 16: Deferred Choice
Execute one of two alternative threads.
The choice [...] should be implicit.



10

Calculus of Communicating Systems

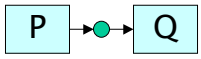
Agenda

- Her bruges følgende variant:

$$P \quad ::= \quad \mathbf{0} \mid \tau.P \mid a?x.P \mid a!x.P \mid P + P \mid (P \mid P) \mid \text{new } a\ P \mid a?*x.P$$

  og atomiske opgaver skrives som *a,b,c* etc.

- **Pattern 1: Sequence**



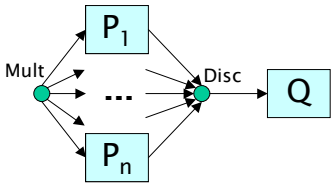
  - a.P – ikke generel nok
  - P.Q – ikke syntaktisk korrekt
  - Vedtag at processer signalerer eksplicit afslutning på, f.eks. *ok*
  - new *go* (P [*go*/*ok*] | *go*?.Q)

11

Eksempelkodninger

Agenda

- **Pattern 7/9: Multiple Choice/Discriminator**
  - ($\tau.P_1$ + τ.skip!) | … | ($\tau.P_n$ + τ.skip!) | ok?.(Q | skip?*.**0** | ok?*.**0**)

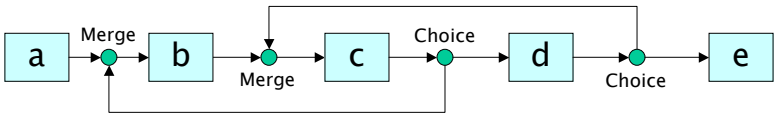


12

Eksempelkodninger

Agenda

- **Pattern 10: Arbitrary Cycles**
  - new go_b, go_c
    ($a.go_b! \mid go_b?^*.b.go_c! \mid$
    $go_c?^*.c.(go_b! + d.(go_c! + e))$



- **Hvad hvis Merge var Sync. Merge?**

13

Hovedobservationer for mønstrene

Agenda

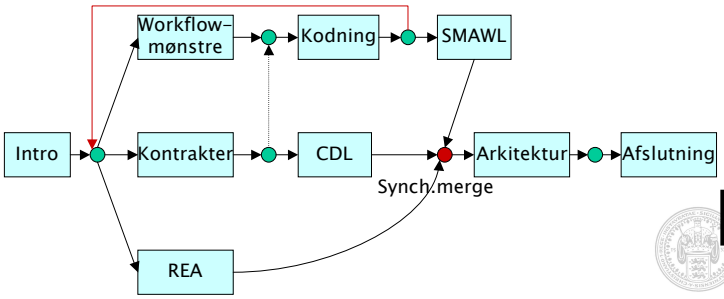
- Split og join er adskilt
- Har fri struktur
- Forsinket vs. øjeblikkeligt valg
- Mange eksotiske split- og join-varianter
- Anullering
- Flere instanser (*Multiple Instances*)

14

Synch. Merge har ikke-lokal semantik

Agenda

- Løsninger
  1. Integrér med split-mønstre
  2. Løs problemet runtime
  3. Modtag metainformation fra tidligere split(s)



15

Flere problemer med mønstrene

Agenda

- Hvad betyder *Discriminator* egentlig?
- Kan man annullere en opgave?
- *Multiple Instances* kombinerer split, join og repetition
- *Synch*, *Merge* eller *Sequence* er overflødig

16

Mønstrene savner formel specifikation

Agenda

- Bedre benchmarking af workflowsprog
- Modeluafhængig (ej Petri net, π, EPC)
- Skal split og join være separate eller ej? (og hvad da med *Multiple Instances*)
- Skal alle mønstre dækkes eller kan data-delen klare nogle af dem?
- Bedre definition af udtrykskraft [Felleisen]

SMAWL – a SMAll Workflow Language

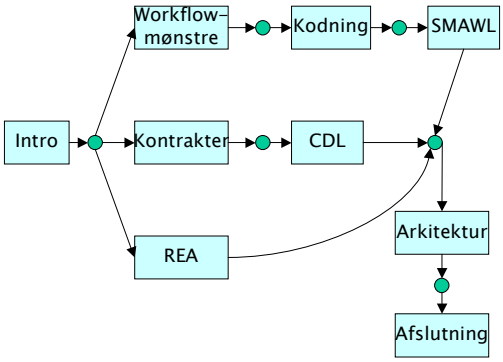
Agenda

- Designkriterier
  - Dæk alle 20 (+2) mønstre (og mere!)
  - Minimér behov for synkroniseringsprimitiver
  - Bevar forbindelse til CCS
  - Uafhængigt af data-mønstre
- Metode
  - Saml mønstre i få, generelle konstruktorer
  - Beskriv deres oversættelse til CCS

Dagens workflow i SMAWL

Agenda

```
workflow Q =
   Intro;
   choose any {
      ⇒ Workflowmønstre;
        Kodning;
        SMAWL
      ⇒ Kontrakter;
        CDL
      ⇒ REA
   };
   Arkitektur;
   Afslutning
end
```



19

Syntaks for SMAWL

Agenda

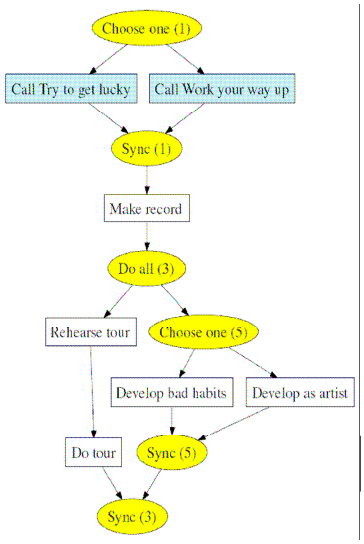
$$
\begin{array}{rcl}
W & ::= & \textbf{workflow}\ w = P\ \textbf{end} \\
D & ::= & \textbf{fun}\ f = P\ \textbf{end}\ D \mid \textbf{newlock}\ (l,u)\ D \\
 & \mid & \textbf{milestone}(ison, isoff, set, clear)\ D \mid \epsilon \\
P & ::= & activity \mid \textbf{send}(f) \mid \textbf{receive}(f) \mid \textbf{call}(f) \mid P;\ P \mid \textbf{lock}(l,u)\{P\} \\
 & \mid & \textbf{choose any (wait for k)}\{PP\ \textbf{merge}(n)\ P\} \mid \textbf{choose one}\{PP\} \mid \textbf{cancel}\ \{P\} \\
 & \mid & \textbf{do all (wait for k)}\{PP\ \textbf{merge}(n)\ P\} \mid \textbf{multi}(n)\{P\} \\
PP & ::= & \Rightarrow \rho\ P\ PP \mid P\ PP \mid \Rightarrow P
\end{array}
$$

20

Britney vil være berømt

Agenda

```
workflow Become a recording star =
    chooseone {
        ⇒ call (Work your way up)
        ⇒ call (Try to get lucky)
    };
    Make record;
    doall {
        ⇒ chooseone {
            ⇒ Develop as an artist
            ⇒ Develop bad habits
            }
        ⇒ Rehearse tour;
            Do tour
    }
end
```



21

Transformation fra SMAWL til CCS

Agenda

- Sequence

$$\mathcal{T}[\![P;Q]\!] = \lambda ok.\text{let } ok' \Leftarrow \nu() \text{ in } \mathcal{T}[\![P]\!]ok' \mid ok'?.\mathcal{T}[\![Q]\!]ok$$

- Multiple Instances

$$\mathcal{T}[\![\textbf{multi}(n)\ \{P\}]\!] = \lambda ok.\text{let } create \Leftarrow \nu() \text{ in let } ok' \Leftarrow \nu() \text{ in}$$
$$\underbrace{create!.\cdots.create!}_{n}.\underbrace{ok'?.\cdots.ok'?}_{n}.ok! \mid create?*.\mathcal{T}[\![P]\!]ok'$$

22

Fremtid

Agenda

- Identificer (flere) svagheder i SMAWL
- Overvej distribution af workflows
- Opstil kriterier for en processmodel og et datasprog
- Formaliser mønstre (om nødvendigt)
- Design kalkule
- Implementér

23

Kommercielle kontrakter

Agenda

- Beskrevet i “Commercial Contracts”
- Fremtiden kunne være...
  - ...valuering à la Black-Scholes
  - ...udnyt forbindelsen til WS-CDL

24

REA og dobbelt bogholderi

Agenda

- Vores datamodel skal opfylde følgende:
  - Transaktionsspor
  - Hændelsesbaseret log (monoton)
  - Alle (økonomiske) hændelser skal have en invers
- Dobbelt bogholderi og REA opfylder disse

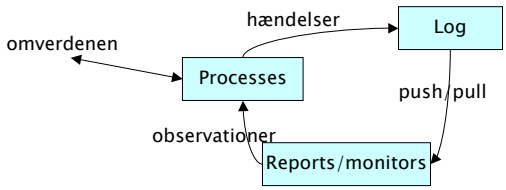
25

Fremtid: et procesbaseret system

Agenda

- Processsprog
- Datasprog
- Servicemønstre
- Auto. GUI-generering

- Datamodel
- Process mining
- Interoperabilitet (ontologi)



- Runtime-verifikation
- Inkrementalisering af rapporter
- Stream processing
- Dynamisk operatorplacering i overlay-netværk

26

Opsummering

Agenda

- Processer
  - Workflowmønstre – *nyttige men upræcise*
  - CCS-kodning – *muligt men rodet*
  - SMAWL – *lovende men ikke færdigt*
  - Kontraktsprog – *klar til kompositionel analyse*
- Datamodel
  - REA – *kræver præcis håndbog*
- Rapportering

27

Tak!

Agenda

- 5 minutters pause
- Q&A

28