Communication, time and data in declarative process models, with applications to run-time adaptation

Thomas T. Hildebrandt
joint work with S. Debois, D. Basin, T. Slaats, R. Mukkamala, M. Marquard,
(and ongoing thoughts with M. Carbone and H. Normann)
IT University of Copenhagen (ITU) Denmark

BETTY meeting
March 18th, 2016

IT UNIVERSITY OF COPENHAGEN

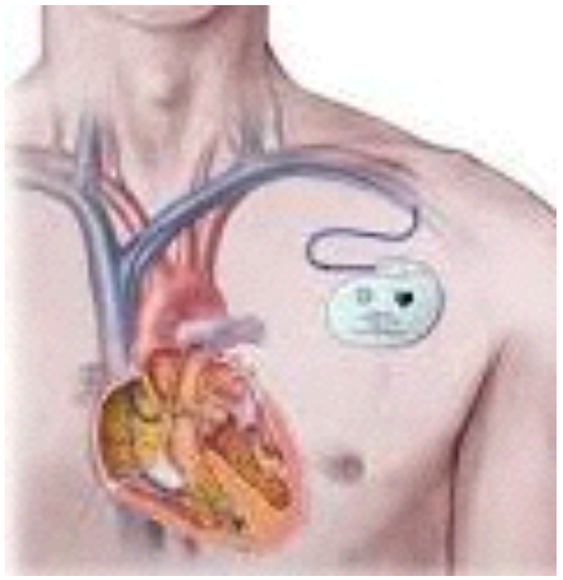
Communication, time and data in declarative process models, with applications to run-time adaptation

or "sequencing and loops considered harmful"

Thomas T. Hildebrandt
joint work with S. Debois, D. Basin, T. Slaats, R. Mukkamala, M. Marquard,
(and ongoing thoughts with M. Carbone and H. Normann)
IT University of Copenhagen (ITU) Denmark

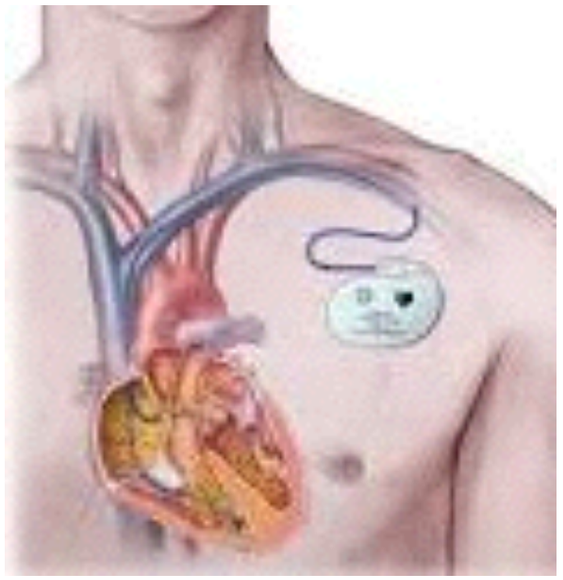
BETTY meeting
March 18th, 2016

IT UNIVERSITY OF COPENHAGEN



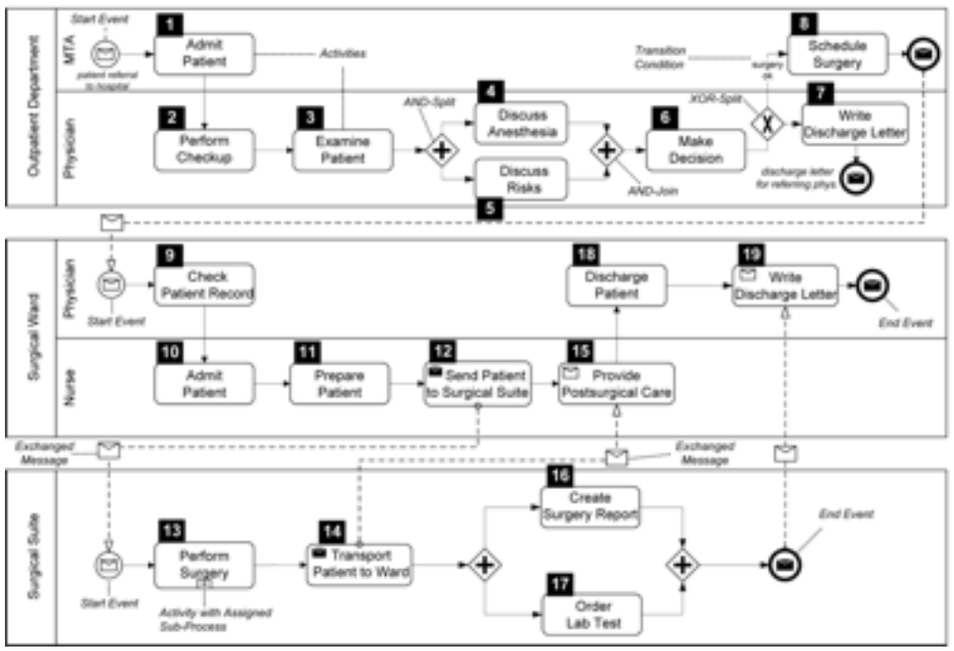
IT systems are increasingly controlling and supporting processes and complex interactions between humans and machines

Thomas T. Hildebrandt



```
                #include "C:\\VCAST\\Tutorial\\cpp\\cpptypes.h"
                #include "C:\\VCAST\\Tutorial\\cpp\\database.h"
                #include "C:\\VCAST\\Tutorial\\cpp\\manager.h"
                void  Manager::PlaceOrder(int Table, int Seat, OrderType Order)
                {
1 0      (T)    Manager::PlaceOrder
                    TableDataType TableData;
                  Data.DeleteTableRecord(&TableData);
                    Data.GetTableRecord(Table, &TableData);
                    TableData.IsOccupied = true;
                    TableData.NumberInParty++;
                    TableData.Order[Seat] = Order;
1 1                 switch (Order.Entree) {
1 2      (T)            case Steak :
                            TableData.CheckTotal += 14;
                            break;
1 4      (T)            case Chicken :
                            TableData.CheckTotal += 10;
                            break;
1 6      (T)            case Lobster :
                            TableData.CheckTotal += 18;
                            break;
1 8      (T)            case Pasta :
                            TableData.CheckTotal += 12;
                            break;
1 10     (T)            default :
                            break;
                    }
```

but the technology is not ready for the complex reality!



Thomas T. Hildebrandt

IT UNIVERSITY OF COPENHAGEN

So we added modularity

Thomas T. Hildebrandt (hilde@itu.dk)

So we added modularity types

So we added modularity
types
exception handling

So we added modularity

types

exception handling

concurrency control & transactions

So we added modularity

types

exception handling

concurrency control & transactions

run-time monitoring

So we added modularity

types

exception handling

concurrency control & transactions

run-time monitoring

and we are adding

So we added modularity

types

exception handling

concurrency control & transactions

run-time monitoring

and we are adding

(imperative) behavioural types

Thomas T. Hildebrandt (hilde@itu.dk)

So we added modularity

types

exception handling

concurrency control & transactions

run-time monitoring

and we are adding

(imperative) behavioural types

run-time adaptation

So we added modularity

types

exception handling

concurrency control & transactions

run-time monitoring

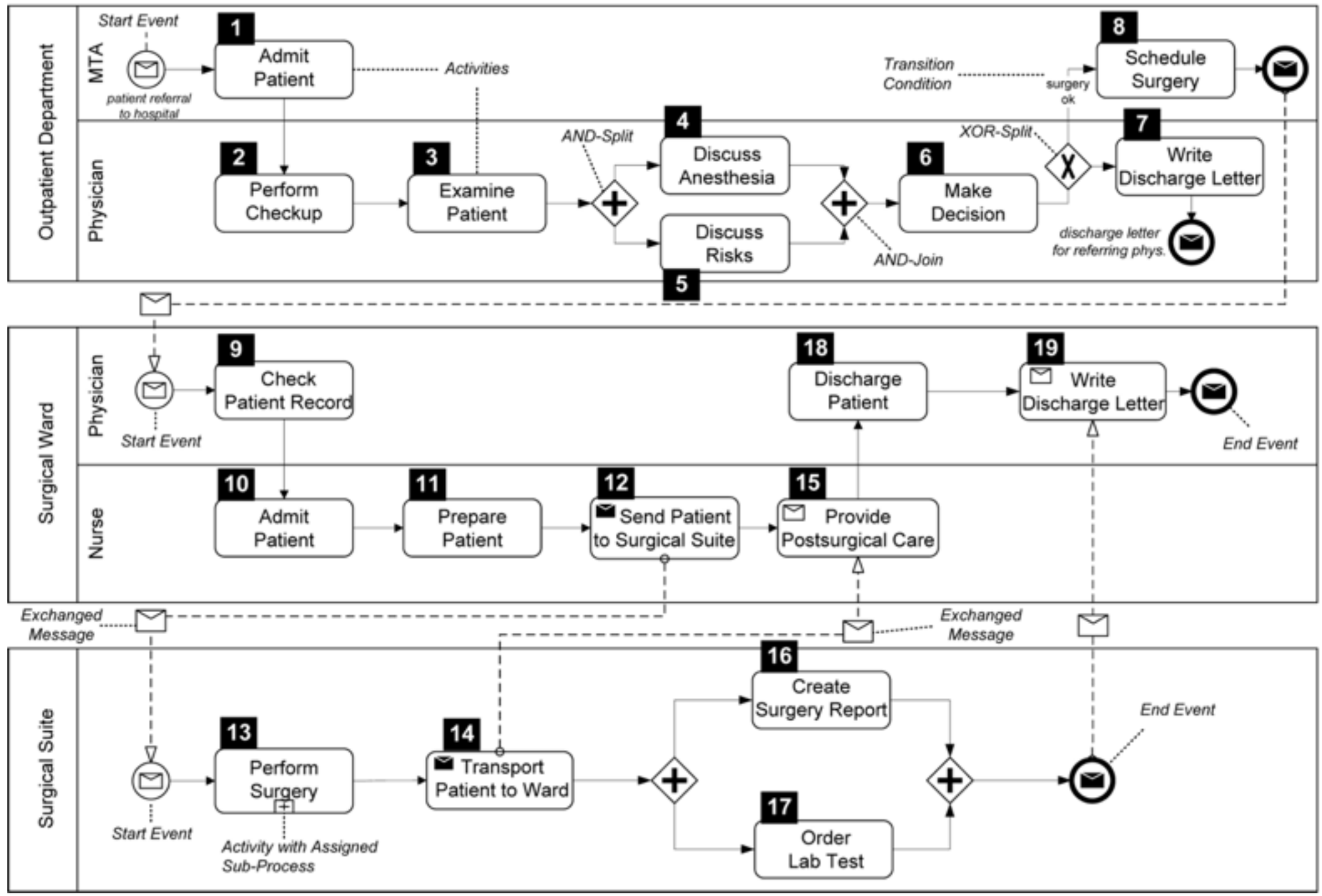
and we are adding

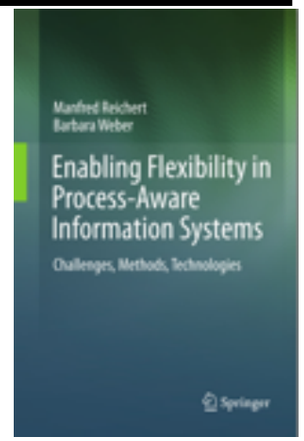
(imperative) behavioural types

run-time adaptation

but we still need to bridge the gap
between requirements (what & why)
& implementation (how)

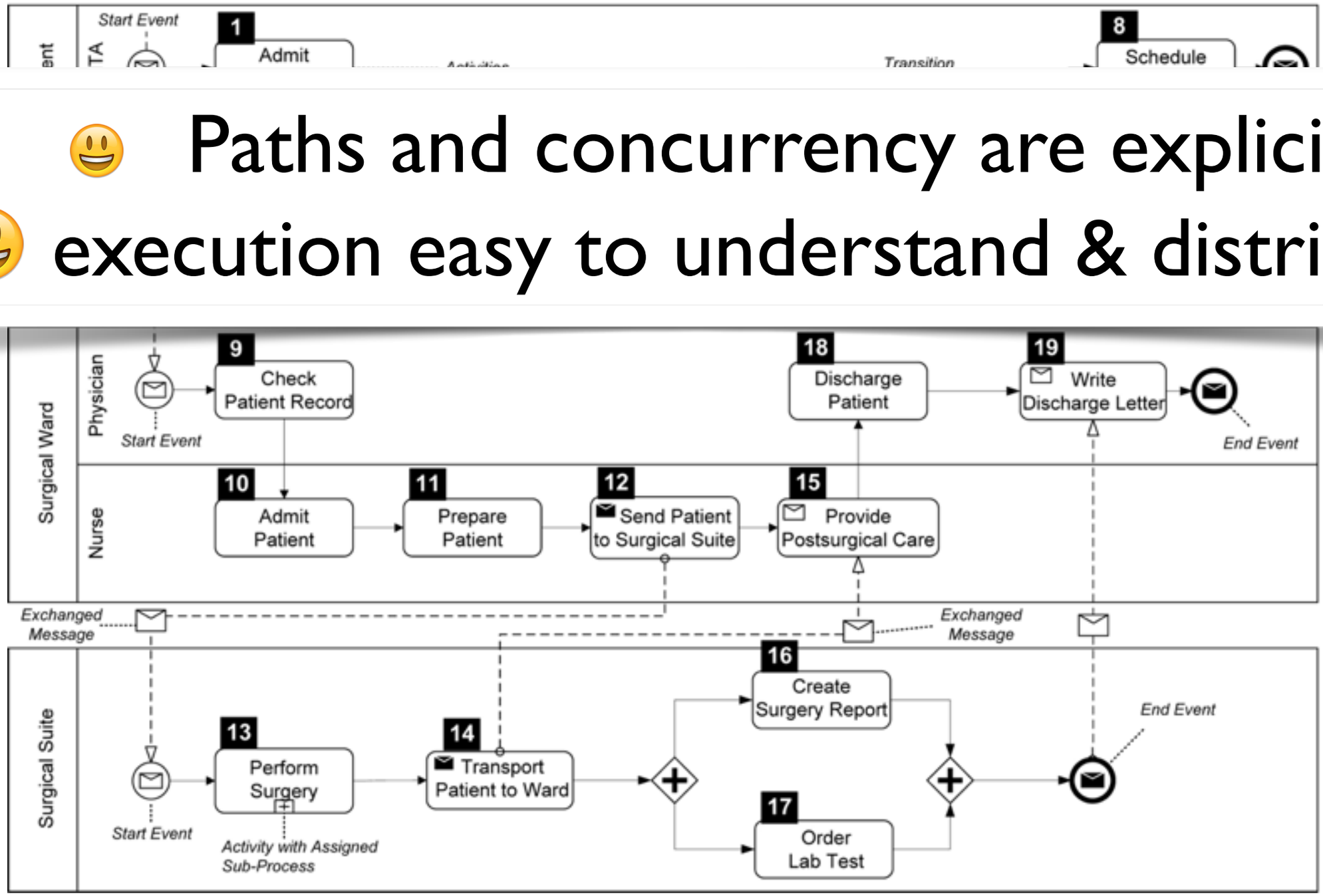
Imperative models prescribe *how*

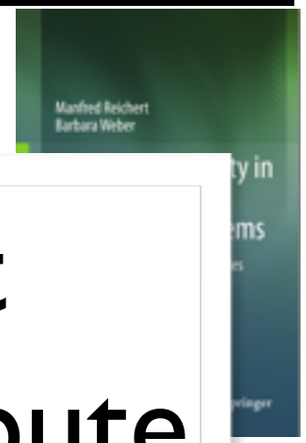




M. Reichert, B.Weber:
Enabling Flexibility in Process-Aware Information Systems.
© Springer-Verlag Berlin Heidelberg 2012

Imperative models prescribe *how*



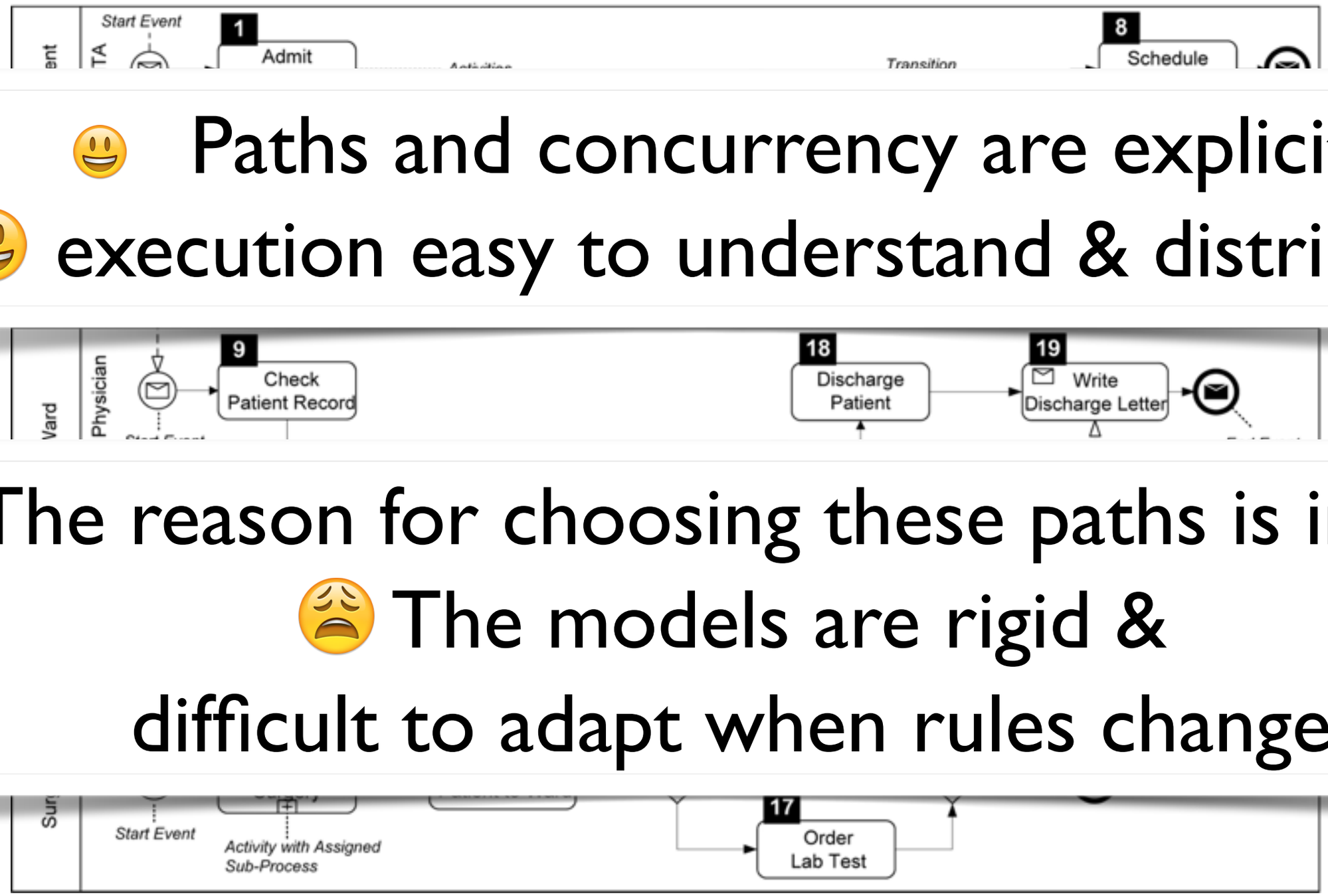


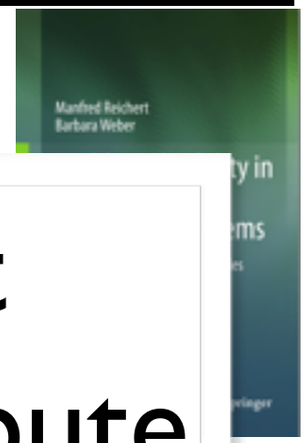
M. Reichert, B.Weber:
Enabling Flexibility in Process-Aware Information
@ Springer-Verlag Berlin Heidelberg 2012

😀 Paths and concurrency are explicit

😀 execution easy to understand & distribute

Imperative models prescribe *how*





😀 Paths and concurrency are explicit

😀 execution easy to understand & distribute

😩 The reason for choosing these paths is implicit

😩 The models are rigid &

difficult to adapt when rules change

Declarative models describe *why*

c_2	After examining the patient a decision must be made. However, this must not be done before the examination.
c_3	After the examination, the patient must be informed about the risks of the (planned) surgery.
c_4	Before scheduling the surgery the patient has to be informed about anesthesia.

Declarative models describe *why*

c_2	After examining the patient a decision must be made. However, this must not be done before the examination.
c_3	After the examination, the patient must be informed about the risks of the (planned) surgery.
c_4	Before scheduling the surgery the patient has to be informed about anesthesia.

conditions

Declarative models describe *why*

c_2	After examining the patient a decision must be made. However, this must not be done before the examination.
c_3	After the examination, the patient must be informed about the risks of the (planned) surgery.
c_4	Before scheduling the surgery the patient has to be informed about anesthesia.

conditions

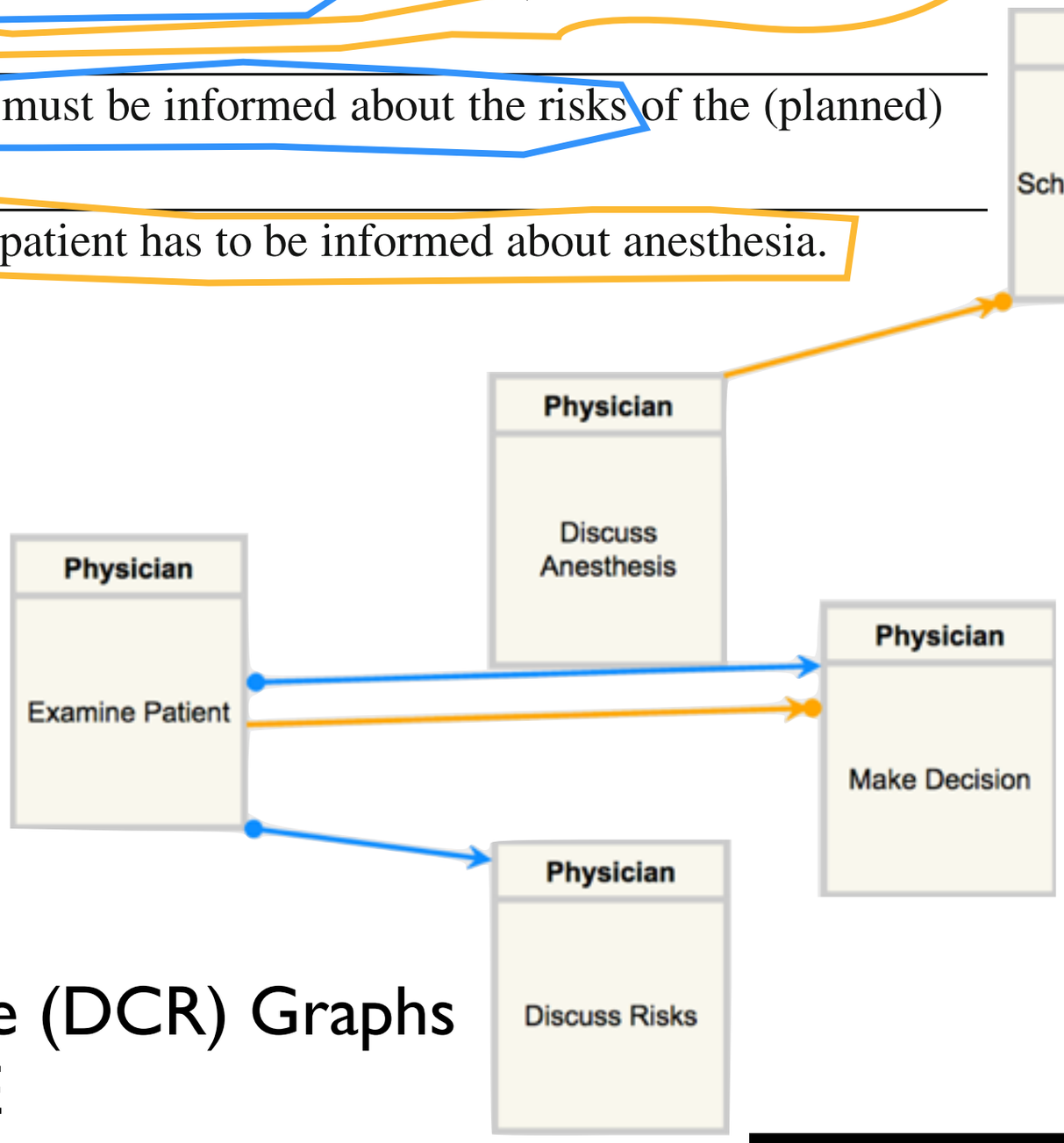
responses

Declarative models describe *why*

c_2	After examining the patient a decision must be made. However, this must not be done before the examination.
c_3	After the examination, the patient must be informed about the risks of the (planned) surgery.
c_4	Before scheduling the surgery the patient has to be informed about anesthesia.

conditions

responses



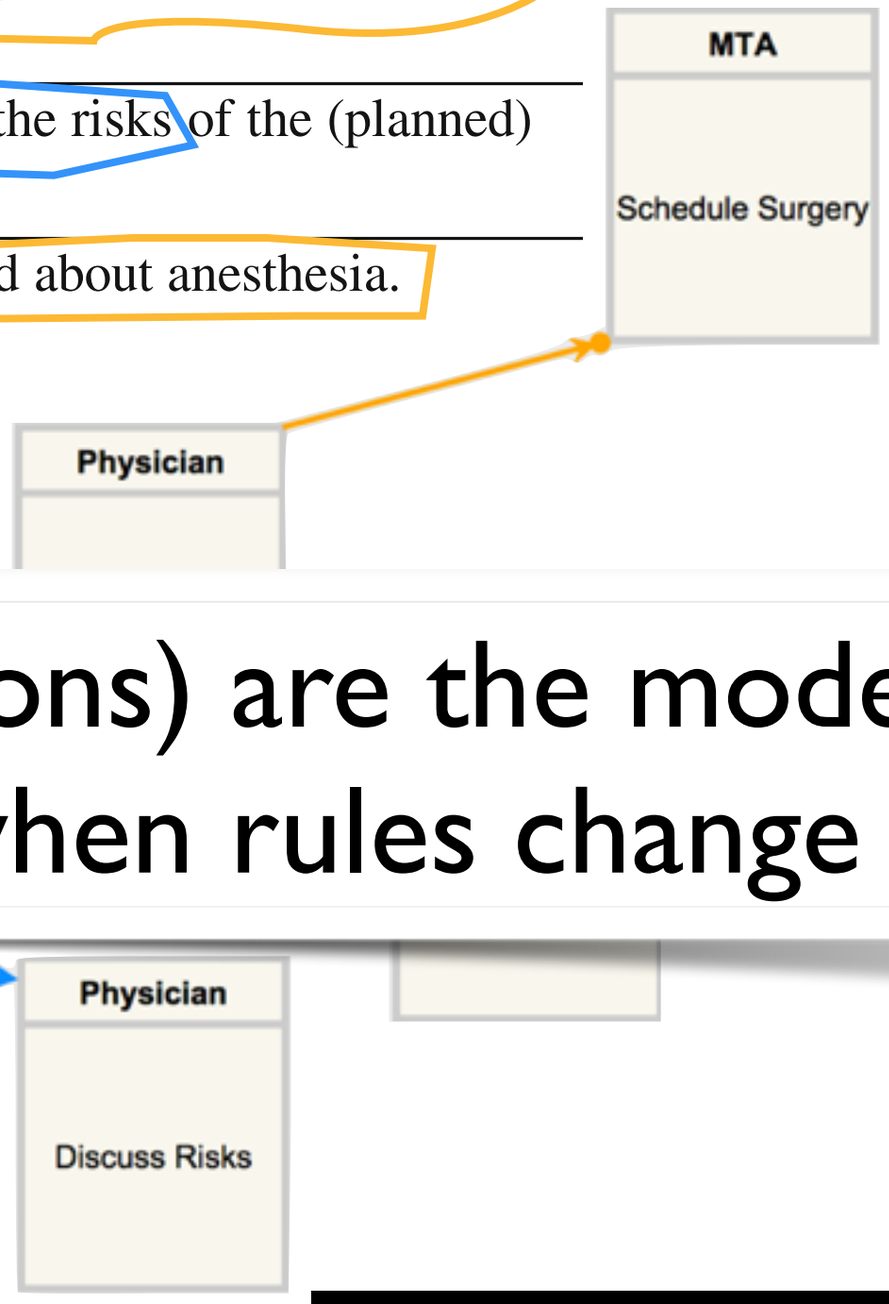
Dynamic Condition Response (DCR) Graphs or DECLARE

Declarative models describe *why*

c_2	After examining the patient a decision must be made. However, this must not be done before the examination.
c_3	After the examination, the patient must be informed about the risks of the (planned) surgery.
c_4	Before scheduling the surgery the patient has to be informed about anesthesia.

conditions

responses

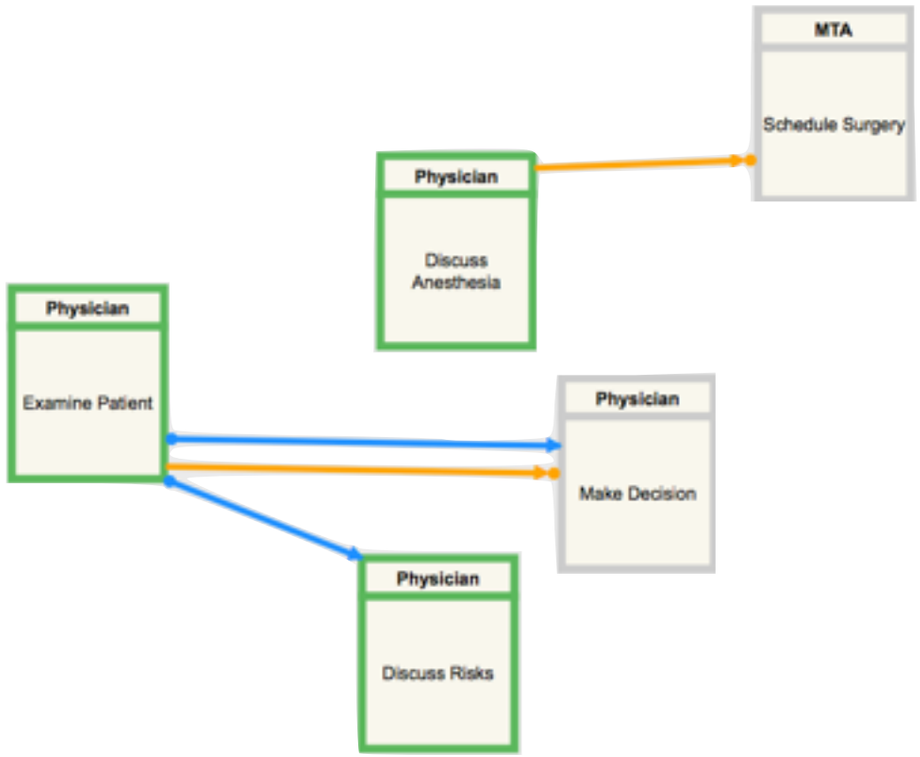


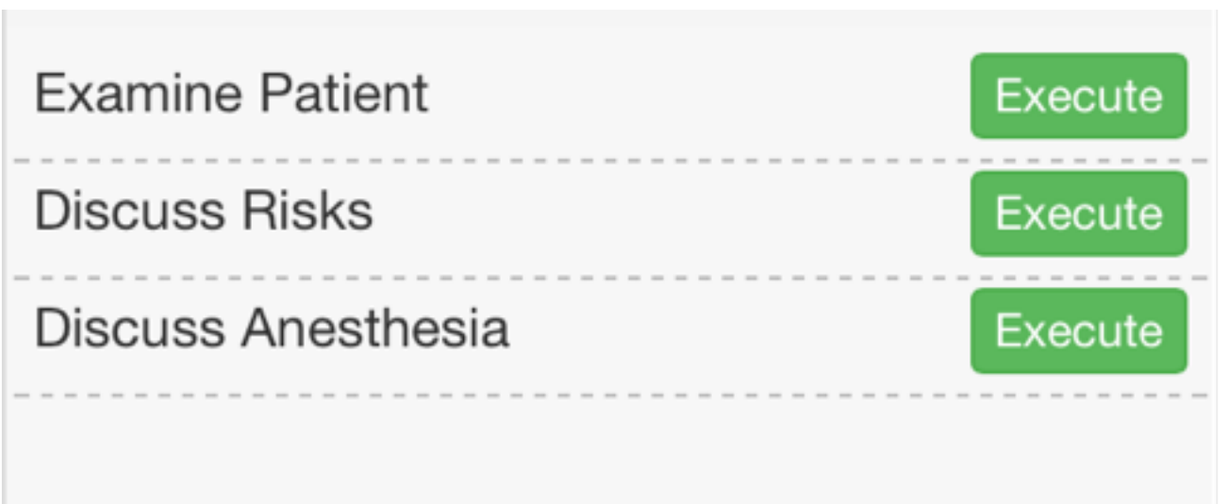
😀 compliance rules (the reasons) are the model
⇒ flexible & maintainable when rules change

Dynamic Condition Response (DCR) Graphs
or DECLARE

Execution of DCR Graphs

eXformatics DCRGraphs.net





Execution of DCR Graphs

eXformatics DCRGraphs.net

Examine Patient	Execute
Discuss Risks	Execute
Discuss Anesthesia	Execute

! Make Decision	Execute
! Discuss Risks	Execute
Discuss Anesthesia	Execute
✓ Examine Patient	Execute

Execution of DCR Graphs

eXformatics DCRGraphs.net

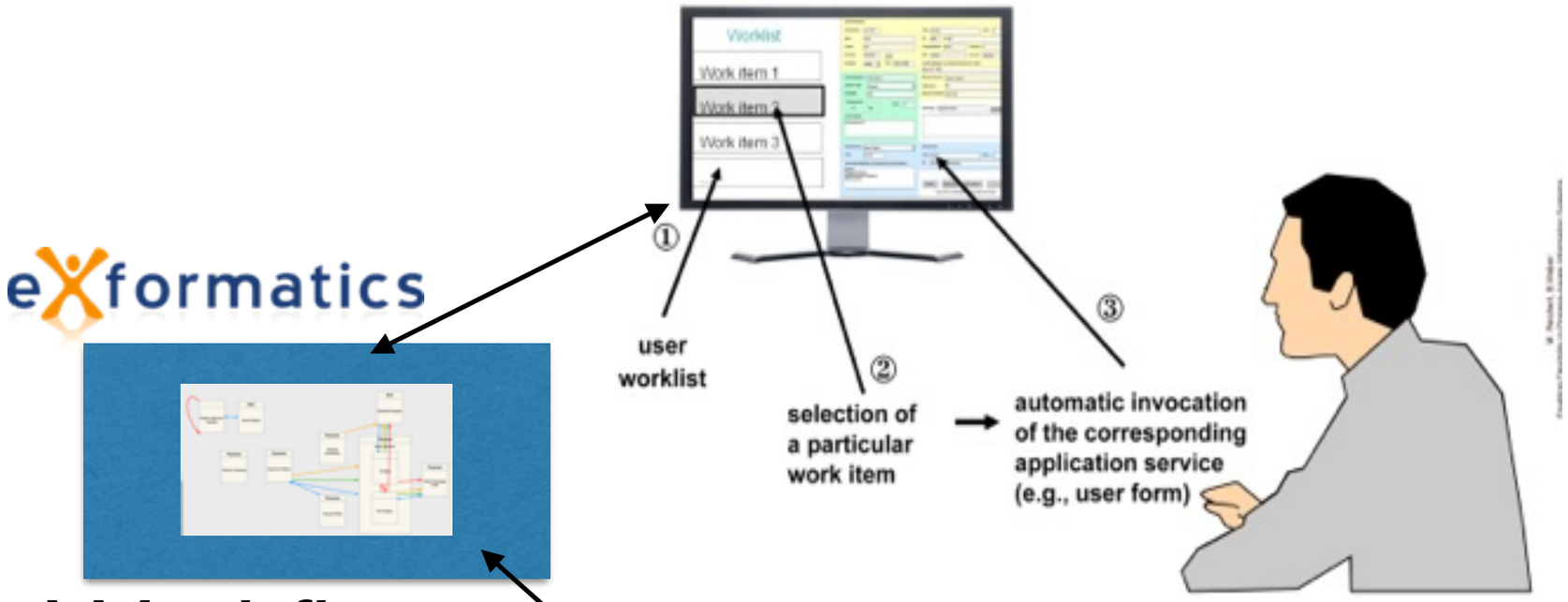
😀 DCR Graphs are directly executable

😀 The state of a graph is a familiar to-do list

Examine Patient	Execute
Discuss Risks	Execute
Discuss Anesthesia	Execute

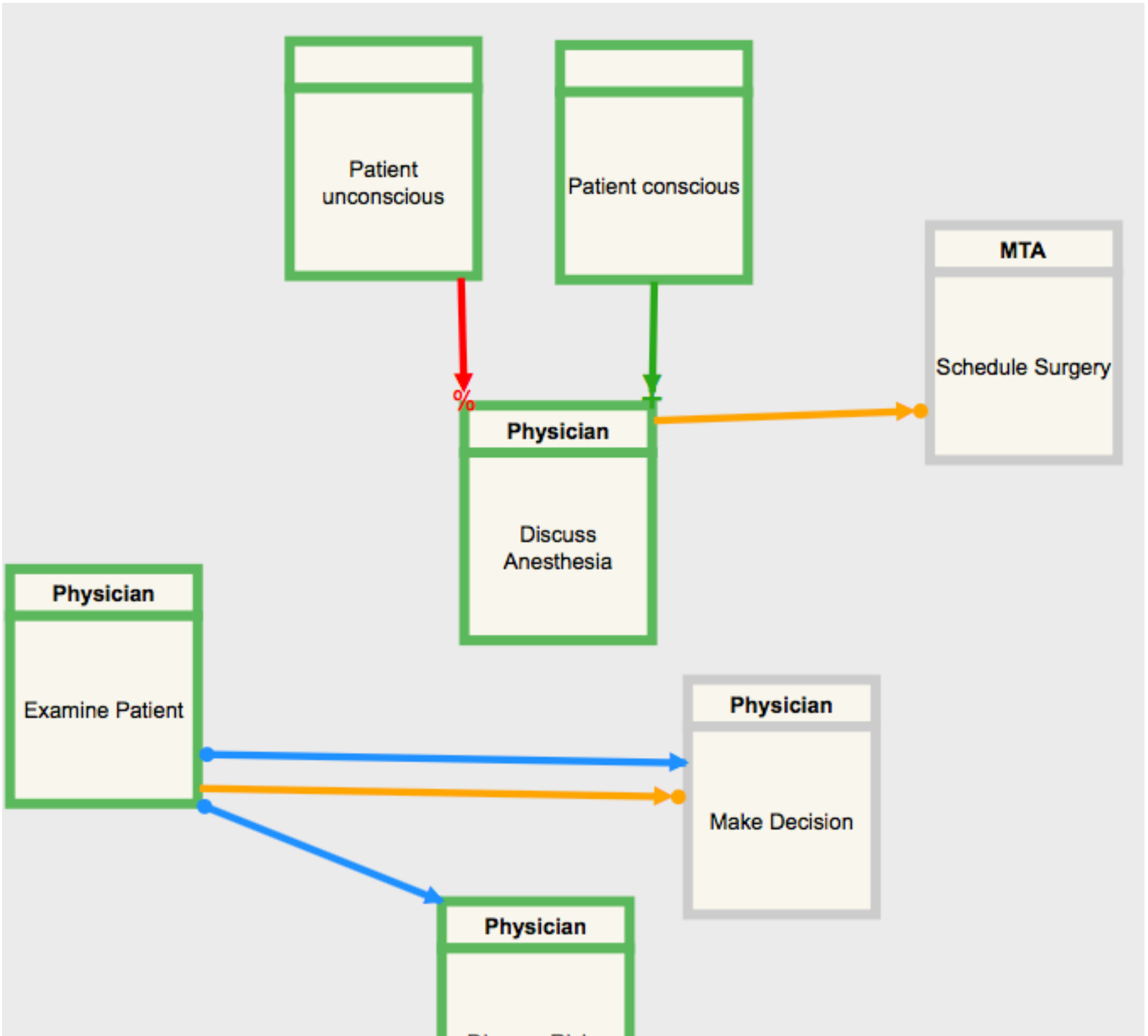
! Make Decision	Execute
! Discuss Risks	Execute
Discuss Anesthesia	Execute
✓ Examine Patient	Execute

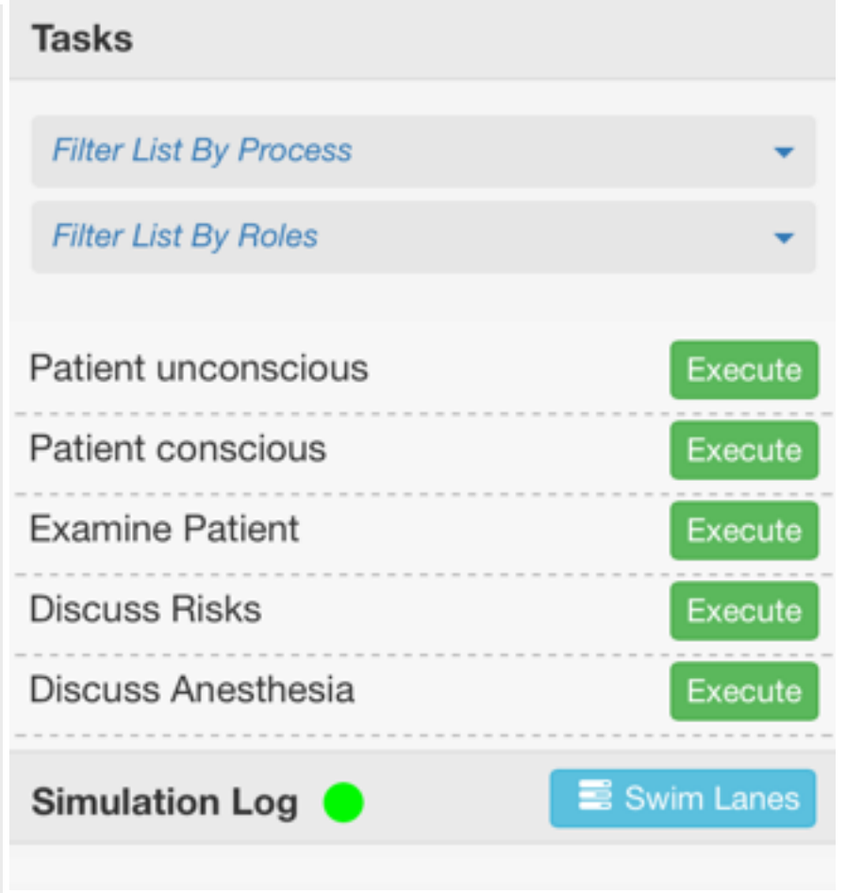
DCR Graphs workflow engine



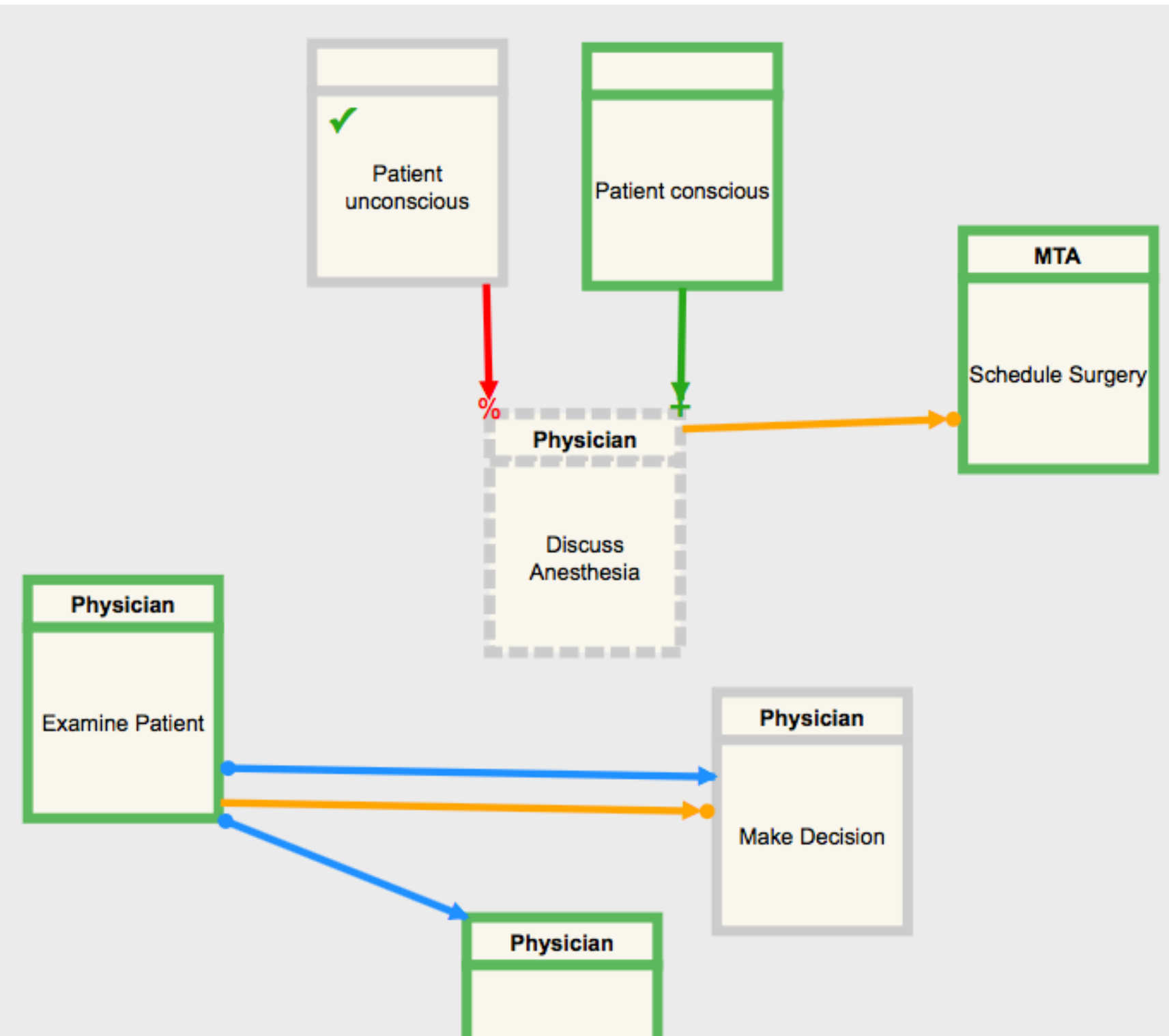
Workflow engine

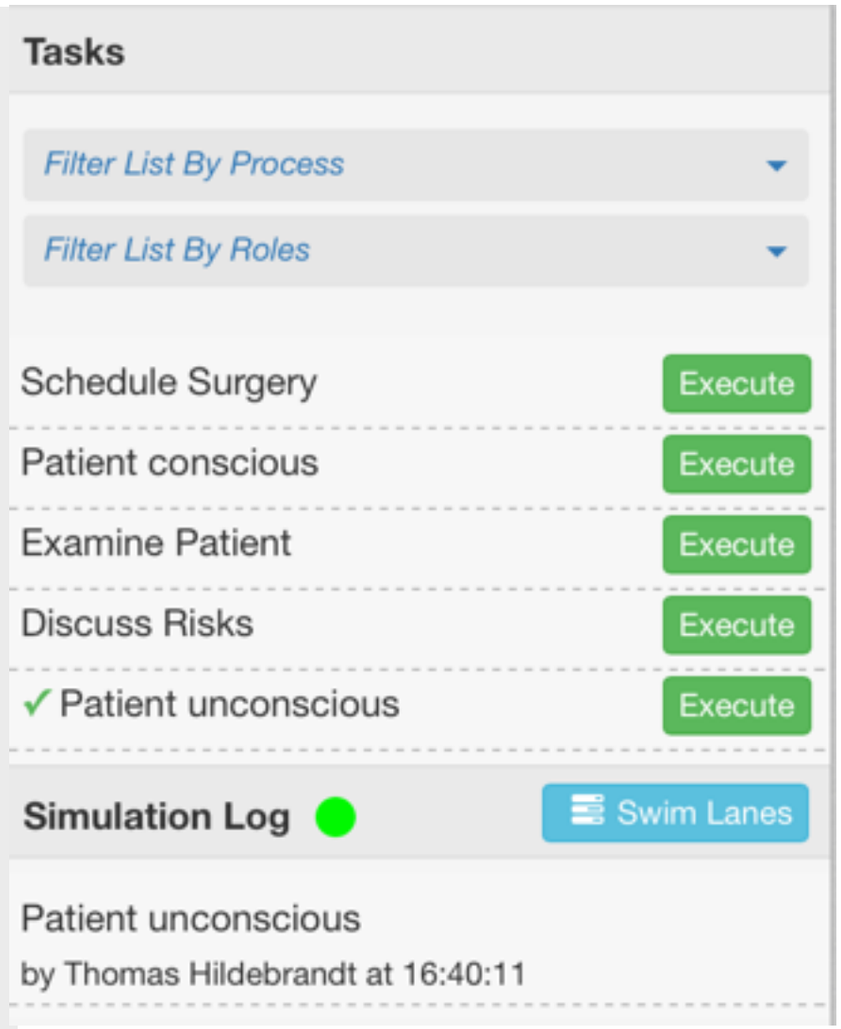
Exclusion and Inclusion in DCR



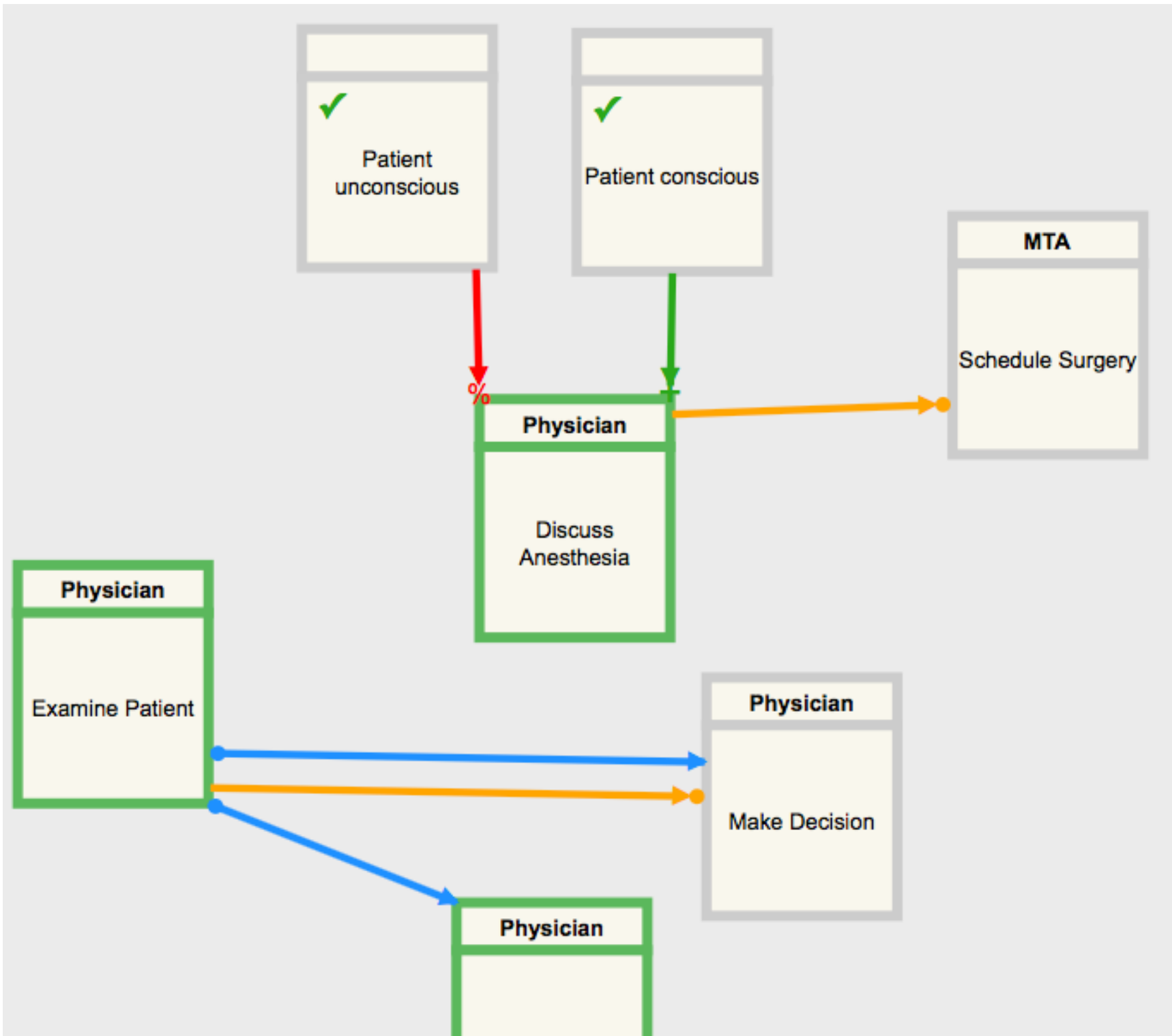


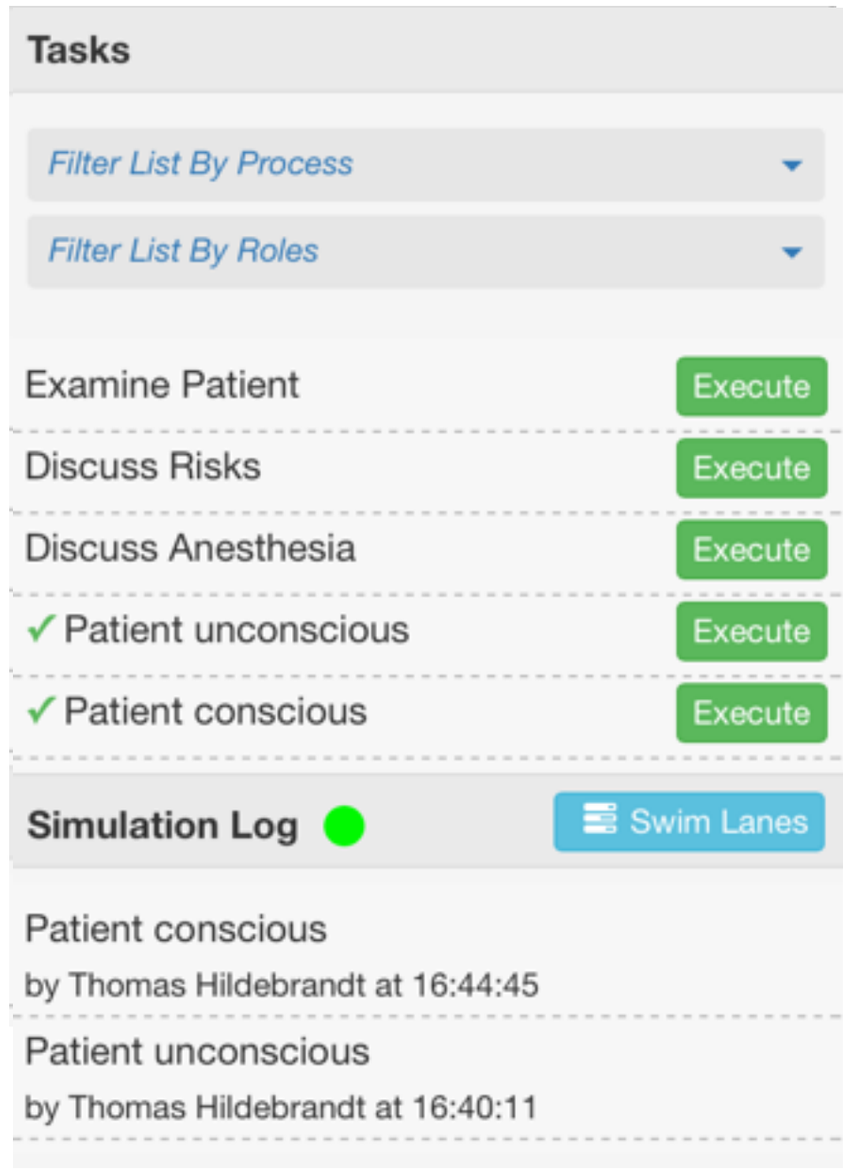
Exclusion and Inclusion in DCR





Exclusion and Inclusion in DCR





A DCR process language

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\leftarrow f & \text{condition} \\
\mid & e \diamond\!\leftarrow f & \text{milestone} \\
\mid & e \bullet\!\rightarrow f & \text{response} \\
\mid & e \rightarrow\!+ f & \text{inclusion} \\
\mid & e \rightarrow\!\% f & \text{exclusion} \\
\mid & T \mid U & \text{parallel} \\
\mid & 0 & \text{unit}
\end{array}
$$

$$
\begin{array}{rll}
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
\mid & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

All regular& omega-regular properties

Implemented at dcr.itu.dk

A DCR process language

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\leftarrow f & \text{condition} \\
| & e \diamond\!\leftarrow f & \text{milestone} \\
| & e \bullet\!\rightarrow f & \text{response} \\
| & e \rightarrow\!+ f & \text{inclusion} \\
| & e \rightarrow\!\% f & \text{exclusion} \\
| & T \mid U & \text{parallel} \\
| & 0 & \text{unit} \\
\\
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
| & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

All regular& omega-regular properties

did the event *h*appen?

Implemented at dcr.itu.dk

A DCR process language

[FM 2015]

$$
\begin{array}{rcll}
T, U & ::= & e \bullet\!\leftarrow f & \text{condition} \\
& | & e \diamond\!\leftarrow f & \text{milestone} \\
& | & e \bullet\!\rightarrow f & \text{response} \\
& | & e \rightarrow\!+ f & \text{inclusion} \\
& | & e \rightarrow\!\% f & \text{exclusion} \\
& | & T \mid U & \text{parallel} \\
& | & 0 & \text{unit} \\
\\
\Phi & ::= & (h, i, r) & \text{event state} \\
M, N & ::= & M, e : \Phi & \text{marking} \\
& | & \epsilon & \\
P, Q & ::= & [M]\, T & \text{process}
\end{array}
$$

All regular& omega-regular properties

did the event *h*appen?

is it *i*ncluded?

Implemented at dcr.itu.dk

A DCR process language

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\leftarrow f & \text{condition} \\
| & e \diamond\!\leftarrow f & \text{milestone} \\
| & e \bullet\!\rightarrow f & \text{response} \\
| & e \rightarrow\!+ f & \text{inclusion} \\
| & e \rightarrow\!\% f & \text{exclusion} \\
| & T \mid U & \text{parallel} \\
| & 0 & \text{unit} \\
\\
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
| & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

All regular& omega-regular properties

did the event *h*appen?

is it *i*ncluded?

is it *r*equired in the future?

Implemented at dcr.itu.dk

DCR.ITU.DK

Parser

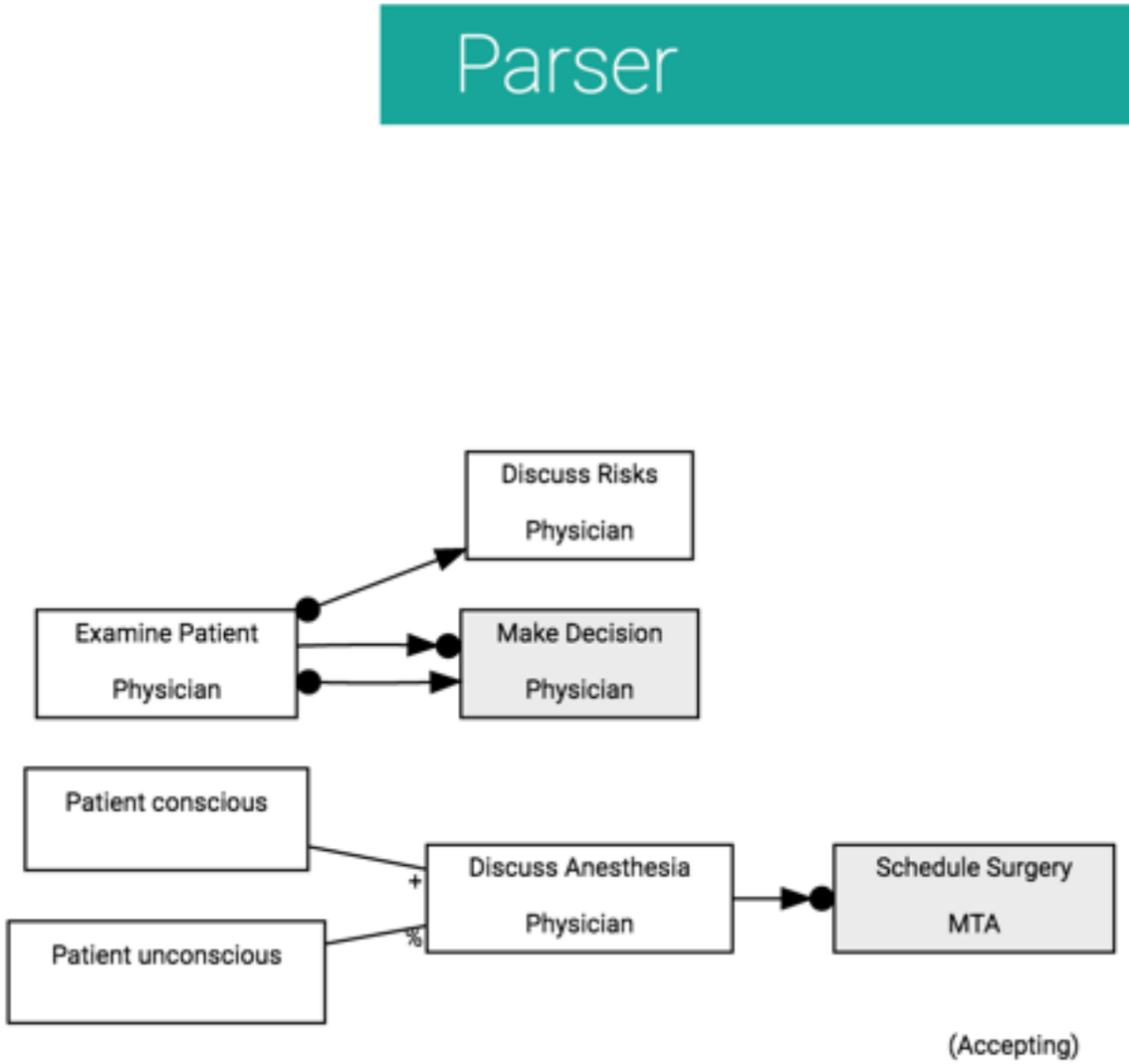
Library

0. Welcome!

Source code

```
"Examine Patient" [role="Physician"]
"Make Decision" [role="Physician"]
"Discuss Risks" [role="Physician"]
"Discuss Anesthesia" [role="Physician"]
"Schedule Surgery" [role="MTA"]
"Patient unconscious"-->%"Discuss Anesthesia"
"Patient conscious"-->+"Discuss Anesthesia"
"Discuss Anesthesia"-->* "Schedule Surgery"
"Examine Patient" *--> "Make Decision"
"Examine Patient" -->* "Make Decision"
"Examine Patient" *--> "Discuss Risks"
```

MERGE LOAD



Semantics

$$[M, f : (h, i, \_), e : (\_, \mathtt{t}, \_)]\ f \rightarrow\!\bullet\ e \vdash e : \emptyset, \emptyset, \emptyset \qquad (\text{when } i \Rightarrow h)$$

$$[M, e : (\_, \mathtt{t}, \_)]\ f \leftarrow\!\bullet\ e \vdash e : \emptyset, \emptyset, \{f\}$$

$$[M, e : (\_, \mathtt{t}, \_)]\ f +\!\leftarrow\ e \vdash e : \emptyset, \{f\}, \emptyset$$

$$[M, e : (\_, \mathtt{t}, \_)]\ f\ \%\!\leftarrow\ e \vdash e : \{f\}, \emptyset, \emptyset$$

$$[M, e : (\_, \mathtt{t}, \_)]\ 0 \vdash e : \emptyset, \emptyset, \emptyset$$

$$[M, e : (\_, \mathtt{t}, \_)]\ f' \mathcal{R}\ f \vdash e : \emptyset, \emptyset, \emptyset \qquad (\text{when } e \neq f)$$

$$\frac{[M]\ T \vdash e : \delta}{[M]\ T \xrightarrow{e:\delta} T}\ [\textsc{intro}] \qquad \frac{[M]\ T_1 \xrightarrow{e:\delta_1} T_1' \qquad [M]\ T_2 \xrightarrow{e:\delta_2} T_2'}{[M]\ T_1 \mid T_2 \xrightarrow{e:\delta_1 \oplus \delta_2} T_1' \mid T_2'}\ [\textsc{par}]$$

$$\frac{[M]\ T \xrightarrow{e:\delta} T'}{[M]\ T \xrightarrow{e} [e : \delta \cdot M]\ T'}\ [\textsc{effect}]$$

DCR* - sub processes/replication

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\leftarrow f & \text{condition} \\
| & e \diamond\!\leftarrow f & \text{milestone} \\
| & e \bullet\!\rightarrow f & \text{response} \\
| & e \rightarrow\!+ f & \text{inclusion} \\
| & e \rightarrow\!\% f & \text{exclusion} \\
| & T \mid U & \text{parallel} \\
| & 0 & \text{unit} \\
| & (\nu e : \Phi)\, T & \text{local event} \\
| & e\{T\} & \text{reproductive event}
\end{array}
$$

$$
\frac{[M]\, T' \xrightarrow{e:\delta} T'' \quad T \cong_\alpha T'}{[M]\, e\{T\} \xrightarrow{e:\delta} e\{T\} \mid T''} \quad [\text{REP}]
$$

$$
\begin{array}{rll}
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
| & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

Acceptance undecidable

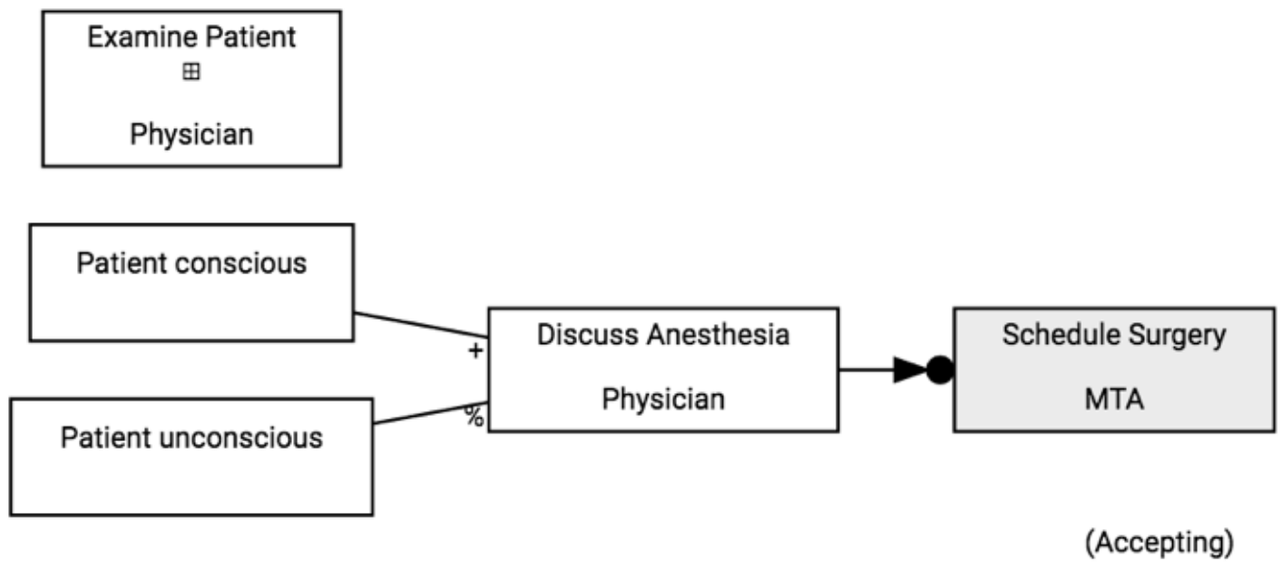
also implemented at dcr.itu.dk

Sub process example

Source code

```
"Examine Patient" [role="Physician"]
"Discuss Anesthesia" [role="Physician"]
"Schedule Surgery" [role="MTA"]
"Patient unconscious"-->%"Discuss Anesthesia"
"Patient conscious"-->+"Discuss Anesthesia"
"Discuss Anesthesia"-->* "Schedule Surgery"


"Examine Patient"{
/!"Make Decision" [role="Physician"]
/!"Discuss Risks" [role="Physician"]
"Discuss Risks"-->*"Make Decision"
"Make Decision"-->%"Make Decision"
"Make Decision"-->%"Discuss Risks"
}
```



Alternating bit-protocol

the Alternating Bit Protocol. Alice repeatedly sends to Bob alternating messages *Msg*1 and *Msg*2

```
!"Msg1" [role="Alice"]          #Message 1 is initially required
"Msg2"  [role="Alice"]
"Msg1"*-->"Msg2"   #After Msg 1 eventually Msg2 must be send
"Msg2"*-->"Msg1"   #After Msg 2 eventually Msg1 must be send
"Msg2"--<>"Msg1"    #Msg1 can not be send while Msg2 is required"
"Msg1"--<>"Msg2"    #Msg2 can not be send while Msg1 is required"
```

Multiparty Session Types Meet Communicating Automata

Pierre-Malo Deniélou and Nobuko Yoshida

Department of Computing, Imperial College London

Alternating bit-protocol

the Alternating Bit Protocol. Alice repeatedly sends to Bob alternating messages $Msg1$ and $Msg2$

```
!"Msg1" [role="Alice"]          #Message 1 is initially required
"Msg2"  [role="Alice"]
"Msg1"*-->"Msg2"   #After Msg 1 eventually Msg2 must be send
"Msg2"*-->"Msg1"   #After Msg 2 eventually Msg1 must be send
"Msg2"--<>"Msg1"    #Msg1 can not be send while Msg2 is required"
"Msg1"--<>"Msg2"    #Msg2 can not be send while Msg1 is required"
```

Multiparty Session Types Meet Communicating Automata

Pierre-Malo Deniélou and Nobuko Yoshida

Department of Computing, Imperial College London

but will always concurrently wait for the acknowledgement Ack_i to send Msg_i.

```
"Msg1" {
   /!"Ack1" [role="Bob"]
   "Ack1" --<> "Msg1"
   "Ack1"-->%"Ack1" }
```

```
"Msg2" {
   /!"Ack2" [role="Bob"]
   "Ack2" --<> "Msg2"
   "Ack2"-->%"Ack2" }
```

dcr.itu.dk

timed DCR*

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\!\xleftarrow{k} f & \text{condition, } k \in \mathbf{N} \cup \{0\} \\
| & e \diamond\!\!\leftarrow f & \text{milestone} \\
| & e \bullet\!\!\xrightarrow{d} f & \text{response, } d \in \mathbf{N} \cup \{\omega\} \\
| & e \rightarrow\!\!+ f & \text{inclusion} \\
| & e \rightarrow\!\!\% f & \text{exclusion} \\
| & T \mid U & \text{parallel} \\
| & 0 & \text{unit}
\end{array}
$$

$$
\begin{array}{rll}
| & (\nu e : \Phi)\, T & \text{local event} \\
| & e\{T\} & \text{reproductive event}
\end{array}
$$

$$
\begin{array}{rll}
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
| & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

timed DCR*

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\!\xleftarrow{k} f & \text{condition, } k \in \mathbf{N} \cup \{0\} \\
\mid & e \diamond\!\!\leftarrow f & \text{milestone} \\
\mid & e \bullet\!\!\xrightarrow{d} f & \text{response, } d \in \mathbf{N} \cup \{\omega\} \\
\mid & e \rightarrow\!\!+ f & \text{inclusion} \\
\mid & e \rightarrow\!\!\% f & \text{exclusion} \\
\mid & T \mid U & \text{parallel} \\
\mid & 0 & \text{unit}
\end{array}
$$

$$
\begin{array}{rll}
\mid & (\nu e : \Phi)\, T & \text{local event} \\
\mid & e\{T\} & \text{reproductive event}
\end{array}
$$

how long ago did the event *h*appen?

$$
\begin{array}{rll}
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
\mid & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

timed DCR*

[FM 2015]

$$
\begin{array}{rll}
T, U ::= & e \bullet\!\!\xleftarrow{k} f & \text{condition, } k \in \mathbf{N} \cup \{0\} \\
| & e \diamond\!\!\leftarrow f & \text{milestone} \\
| & e \bullet\!\!\xrightarrow{d} f & \text{response, } d \in \mathbf{N} \cup \{\omega\} \\
| & e \rightarrow\!\!+ f & \text{inclusion} \\
| & e \rightarrow\!\!\% f & \text{exclusion} \\
| & T \mid U & \text{parallel} \\
| & 0 & \text{unit}
\end{array}
$$

$$
\begin{array}{rll}
| & (\nu e : \Phi)\, T & \text{local event} \\
| & e\{T\} & \text{reproductive event}
\end{array}
$$

how long ago did the event *h*appen?

$$
\begin{array}{rll}
\Phi ::= & (h, i, r) & \text{event state} \\
M, N ::= & M, e : \Phi & \text{marking} \\
| & \epsilon & \\
P, Q ::= & [M]\, T & \text{process}
\end{array}
$$

when is it *r*equired in the future?

Timed data protection policy

(also implemented at dcr.itu.dk)

"Release" *-[14]-> "Delete records"

"Release" *--> "Archive records"

"Release" -->+ "Delete records"

"Archive records" --<> "Delete records"

"Re-admit" -->% "Delete records"

"Archive records" -[28]->* "Delete archived records"

Timed data protection policy

(also implemented at dcr.itu.dk)

"Release" *-[14]-> "Delete records"

"Release" *--> "Archive records"

"Release" -->+ "Delete records"

"Archive records" --<> "Delete records"

"Re-admit" -->% "Delete records"

"Archive records" -[28]->* "Delete archived records"

Time

Time state (after Release)

Event	Last executed	Deadline
Archive records	⊥	ω
Delete archived records	⊥	⊥
Delete records	⊥	14
Re-admit	⊥	⊥
Release	0	⊥

Time control

ADVANCE TIME by 1 ticks

Auto-advance every 5 seconds

Timed data protection policy

"Release" *-[14]-> "Delete records"

"Release" *--> "Archive records"

"Release" -->+ "Delete records"

"Archive records" --<> "Delete records"

"Re-admit" -->% "Delete records"

"Archive records" -[28]->* "Delete archived records"

Time

Time state (after 1 time step)

Event	Last executed	Deadline
Archive records	⊥	ω
Delete archived records	⊥	⊥
Delete records	⊥	13
Re-admit	⊥	⊥
Release	1	⊥

Time control

ADVANCE TIME by 1 ticks

Auto-advance every 5 seconds

Timed data protection policy

"Release" *-[14]-> "Delete records"

"Release" *--> "Archive records"

"Release" -->+ "Delete records"

"Archive records" --<> "Delete records"

"Re-admit" -->% "Delete records"

"Archive records" -[28]->* "Delete archived records"

Time

Time state

(after 14 time steps)

Event	Last executed	Deadline
Archive records	⊥	ω
Delete archived records	⊥	⊥
Delete records	⊥	0
Re-admit	⊥	⊥
Release	14	⊥

Time control

ADVANCE TIME by 1 ticks

Auto-advance every 5 seconds

Advancing time beyond 1 ticks would miss a deadline.

Timed data protection policy

"Release" *-[14]-> "Delete records"

"Release" *--> "Archive records"

"Release" -->+ "Delete records"

"Archive records" --<> "Delete records"

"Re-admit" -->% "Delete records"

"Archive records" -[28]->* "Delete archived records"

Time

Time state

(after 14 time steps)

Event	Last executed	Deadline
Archive records	⊥	ω
Delete archived records	⊥	⊥
Delete records	⊥	0
Re-admit	⊥	⊥
Release	14	⊥

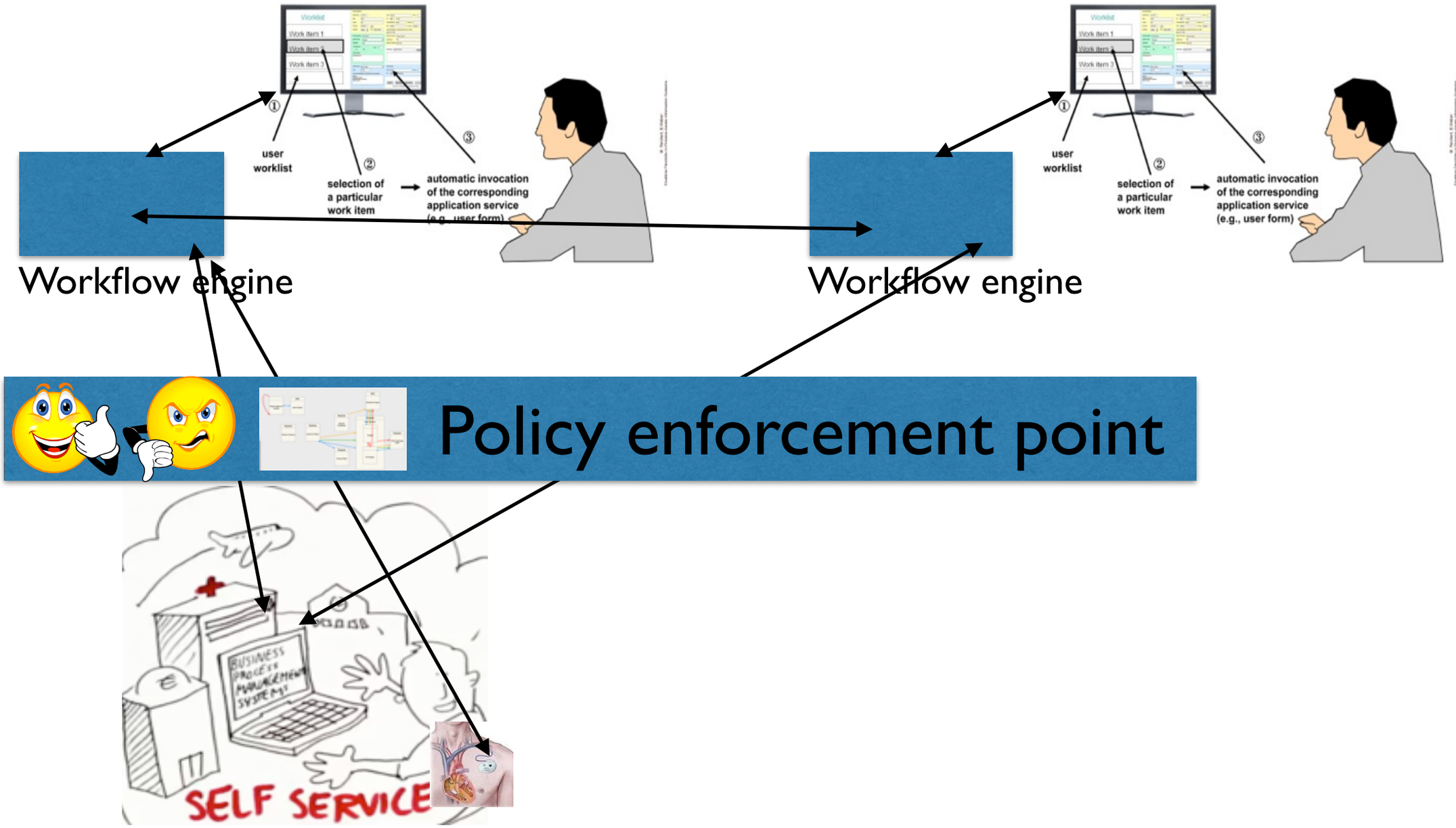
Time control

ADVANCE TIME by 1 ticks

Auto-advance every 5 seconds

Advancing time beyond 1 ticks would miss a deadline.

Policy enforcement



Enforceability & Escalation

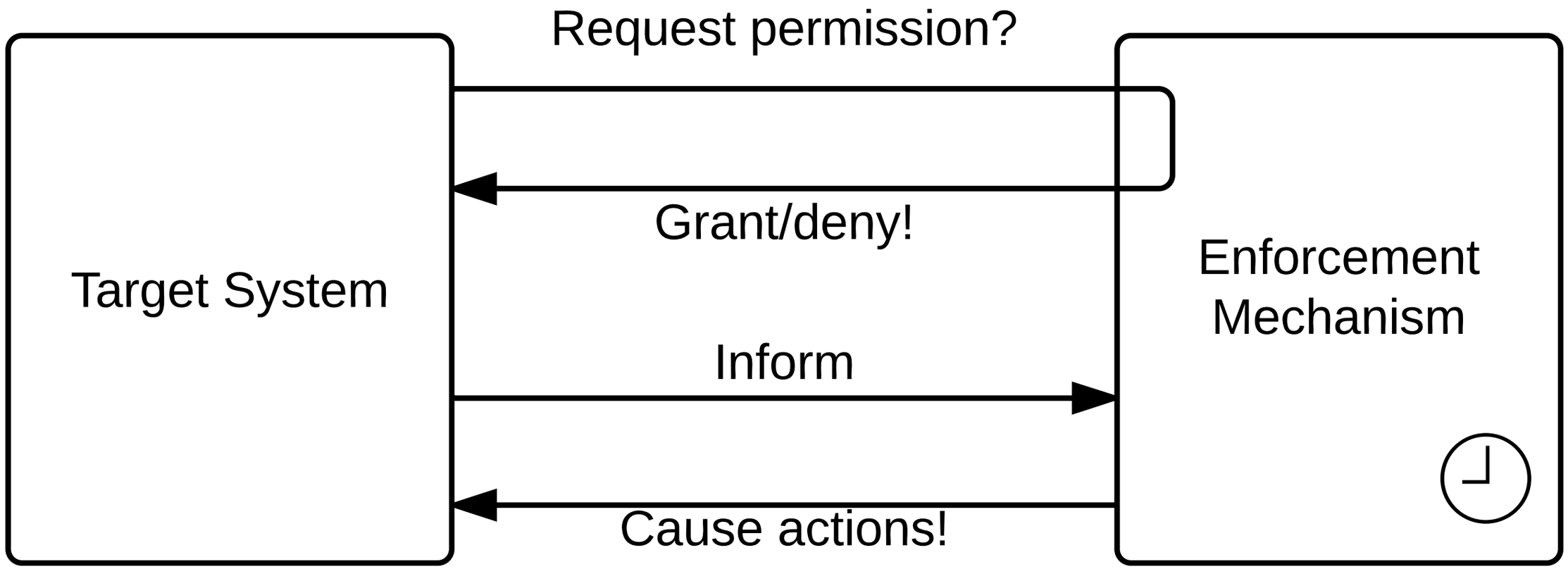
Some events are uncontrollable

in particular human activities

But time is unstoppable….

Need compensation, escalation or proactive enforcement

Proactive Enforcement



Try research-prototype at dcr.itu.dk/obligations

Enforcement

Causability

Event	Causable?
Archive records	✓
Delete archived records	☐
Delete records	✓
Re-admit	☐
Release	☐

Quick selections
All None Invert

Enforcement control

off on

Deadline at 0; taking compensatory action.

Busy events [Lemma 25]

Archive records
Delete records

Inhibition closure of Busy [Defn 29]

Delete records
Archive records

CHOOSE AS CAUSABLE

dcr.itu.dk/obligations

Thomas T. Hildebrandt (hilde@itu.dk)

IT UNIVERSITY OF COPENHAGEN

Implementation in F#

```
// A PEP takes as input an Observation which is
// either an (attempted) transition, or a deadline
// approaching.
type Observation =
  | Transition of DCR.event
  | Deadline   of DCR.event
  | Inform     of DCR.event

// A PEP produces as output a Reaction. (Code for
// acting on the Reaction is not included here.)
type Reaction =
  | Grant
  | Deny
  | Cause of DCR.event list
  | Ignore

// DCR-PEP. Takes a current DCR policy-state P and
// an Observation, and produces a Reaction and a
// new DCR policy-state.
let PEP P observation =
    match observation with
    | Deadline e ->
        Cause (resolve P e), P                  // (i)
    | Transition e ->
        if (DCR.is_executable P e) then
          Grant, DCR.execute e P                // (iii)
        else
          Deny, P                               // (ii)
    | Inform e ->
         Ignore, DCR.execute e P                // (iii)
```

What is special for DCR graphs?

What is special for DCR graphs?

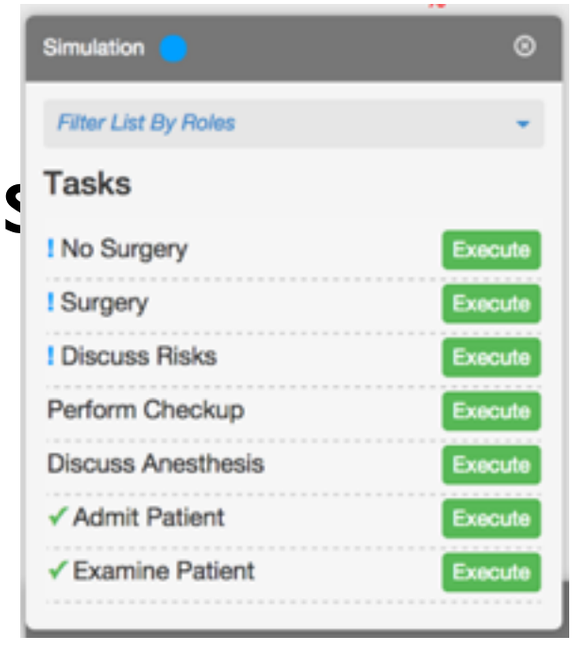
- Formal and close to natural language:
  Conditions, Responses, Inclusions and Exclusions

What is special for DCR graphs?

- Formal and close to natural language:
  Conditions, Responses, Inclusions and Exclusions
- Expressive and decidable:
  Can express all regular safety and liveness properties

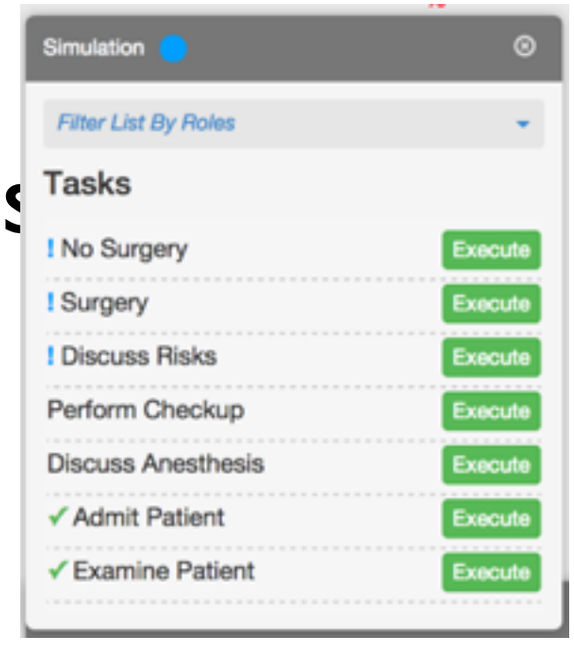
What is special for DCR graphs?

- Formal and close to natural language:
  Conditions, Responses, Inclusions and Exclusions
- Expressive and decidable:
  Can express all regular safety and liveness
- Operational and understandable:
  Run-time state as "check-list" on events



What is special for DCR graphs?

- Formal and close to natural language:
  Conditions, Responses, Inclusions and Exclusions
- Expressive and decidable:
  Can express all regular safety and liveness
- Operational and understandable:
  Run-time state as "check-list" on events
- Efficient monitoring/enactment & adaptable:
  Local decision of enabledness & effect of events



Work so far

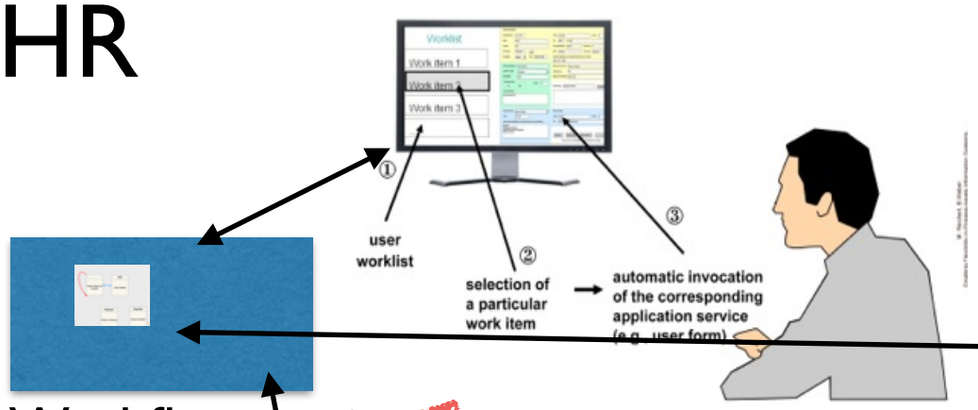
- Tools (DCRGraphs.net, dcr.itu.dk)
- Verification, Time & Dynamic Subprocesses [JLAP82,2013, BPM14,FM15]
- Distribution & Independence [SEFM2011,BPM15]
- Search Path & projections [BPM14]
- Applications to case studies [FHIES2011,ACM14,BPM15]
  (Healthcare, case management & emergency management)
- Run-time adaptation & refinement [EDOC2013][ACM14][FM15]
- Programming Language/Calculi [DEBS2012,REBLS15]

Current activities

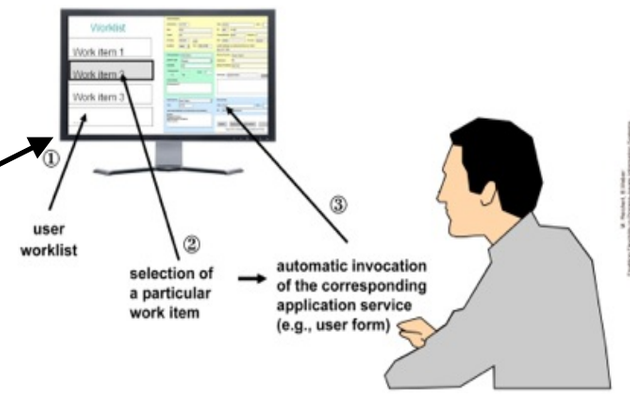
- Case studies
- Extensions (transactional sub processes, data, communication)
- Process Mining & prescriptive process monitoring
- End-point projection
- Supervisory-control-theory

End point projection

HR



FM



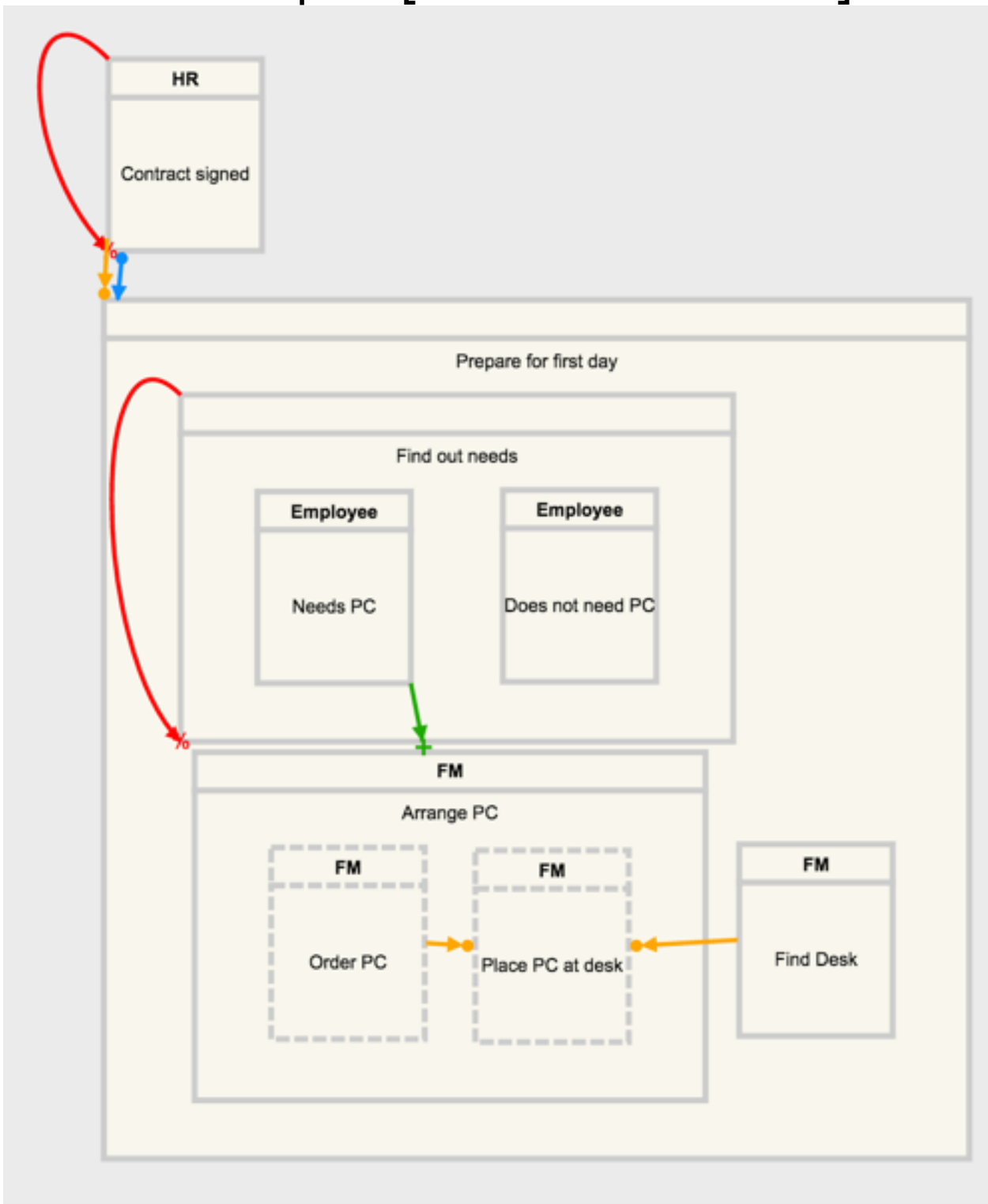
Workflow engine

Workflow engine

Papers: [SEFM2011,FHIES2011]

New employee

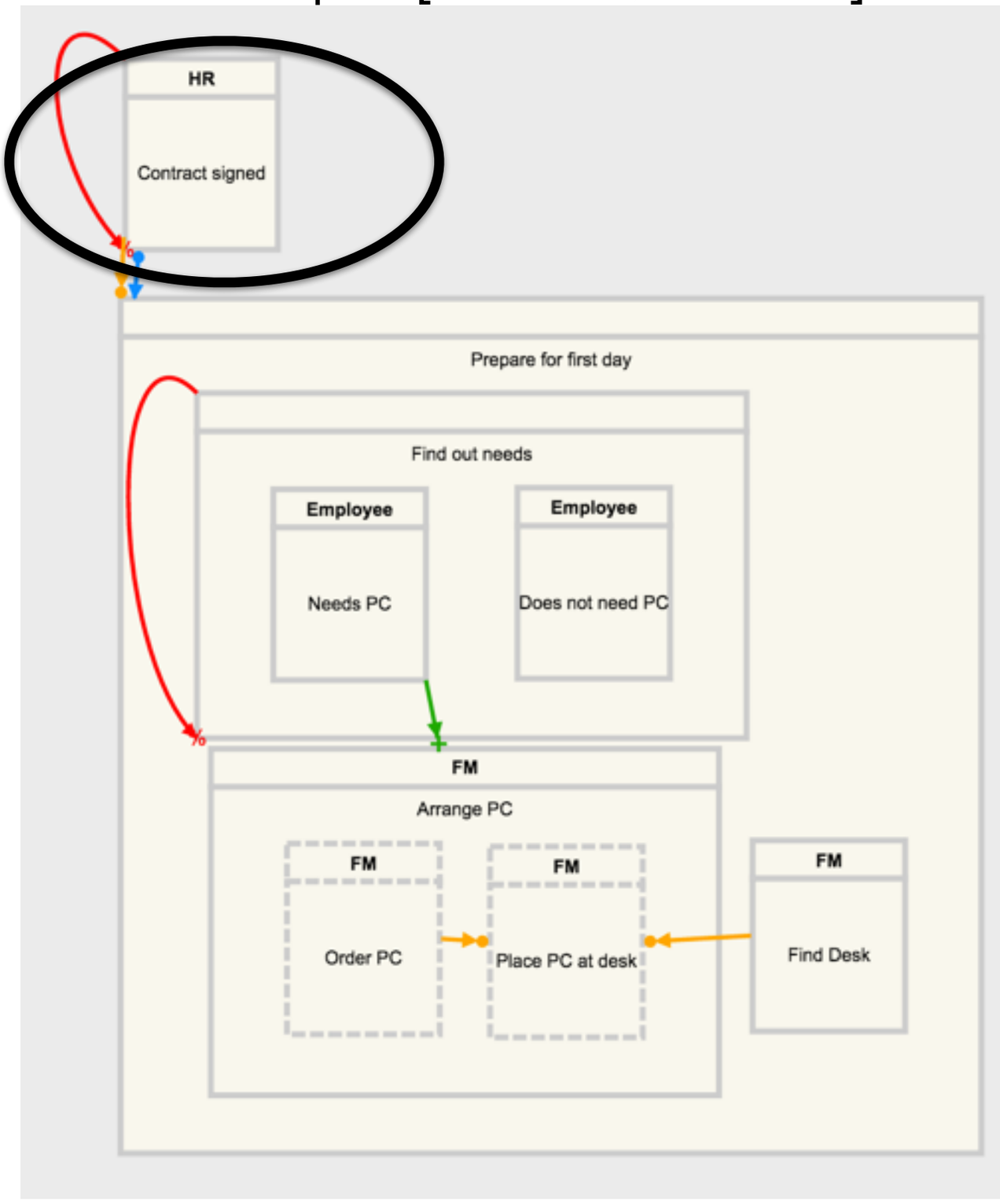
Papers: [SEFM2011,FHIES2011]



Thomas T. Hildebrandt (hilde@itu.dk)

HR

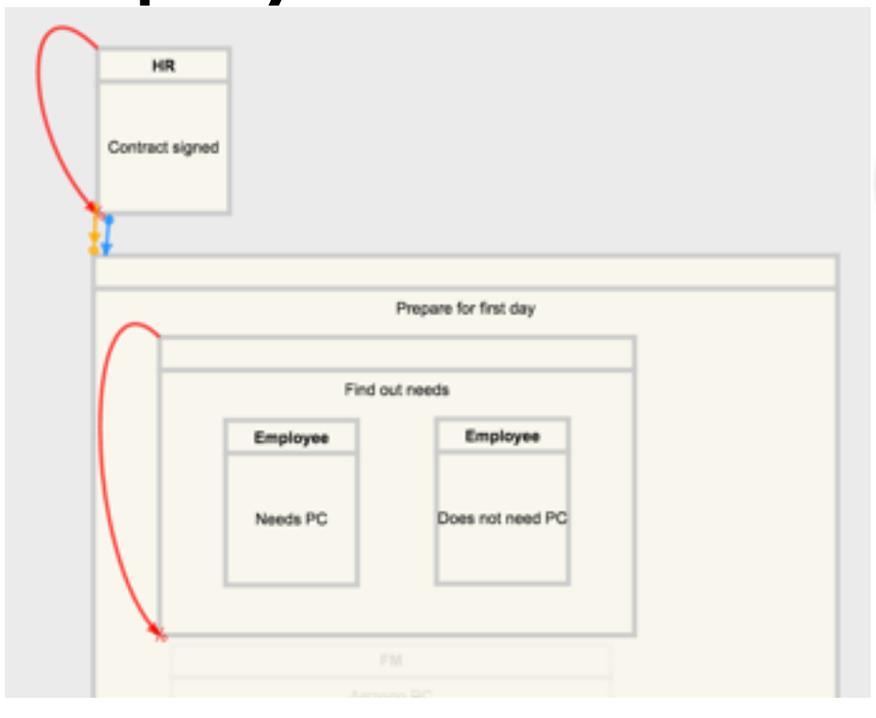
Papers: [SEFM2011,FHIES2011]



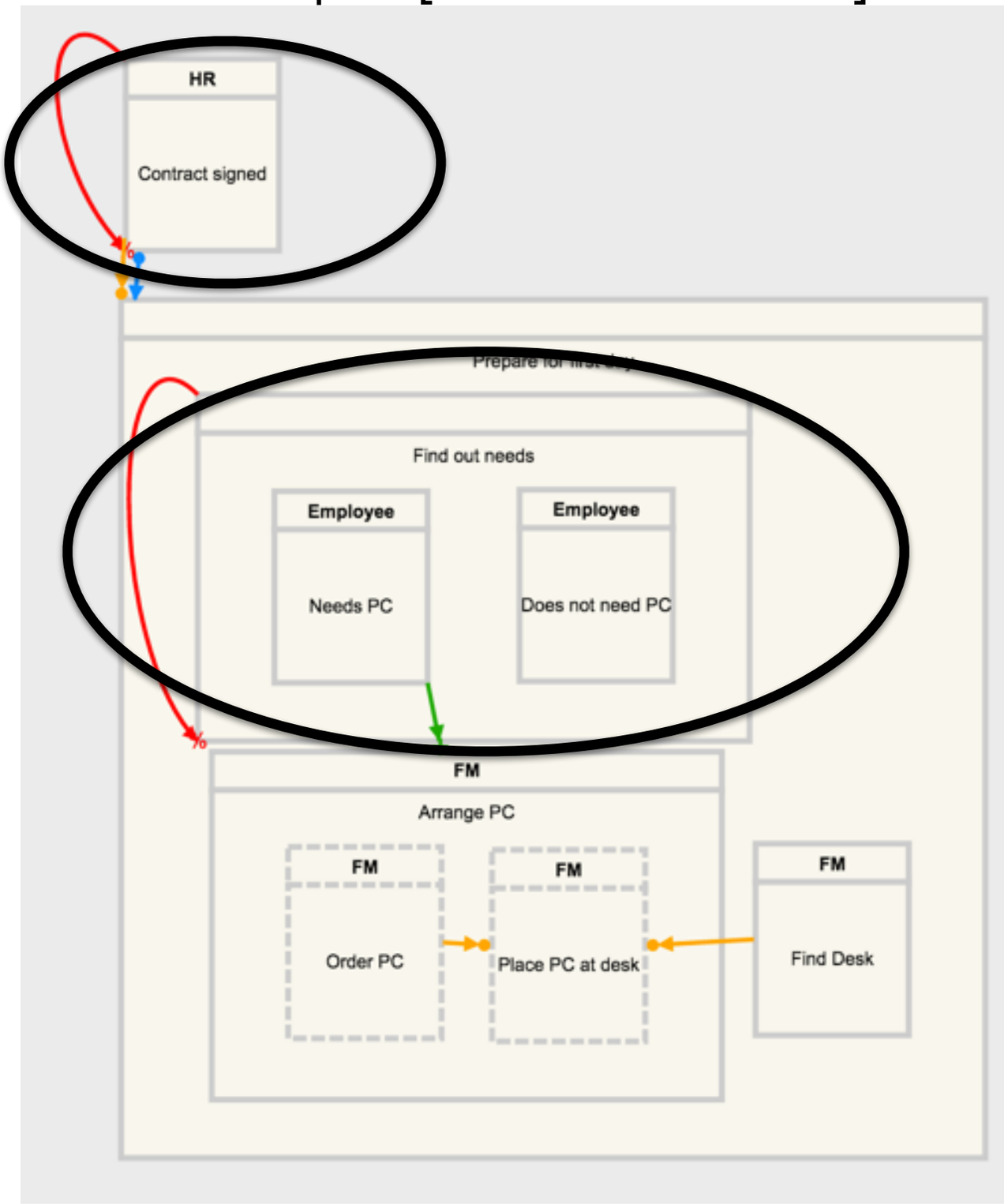
Thomas T. Hildebrandt (hilde@itu.dk)

HR

Employee



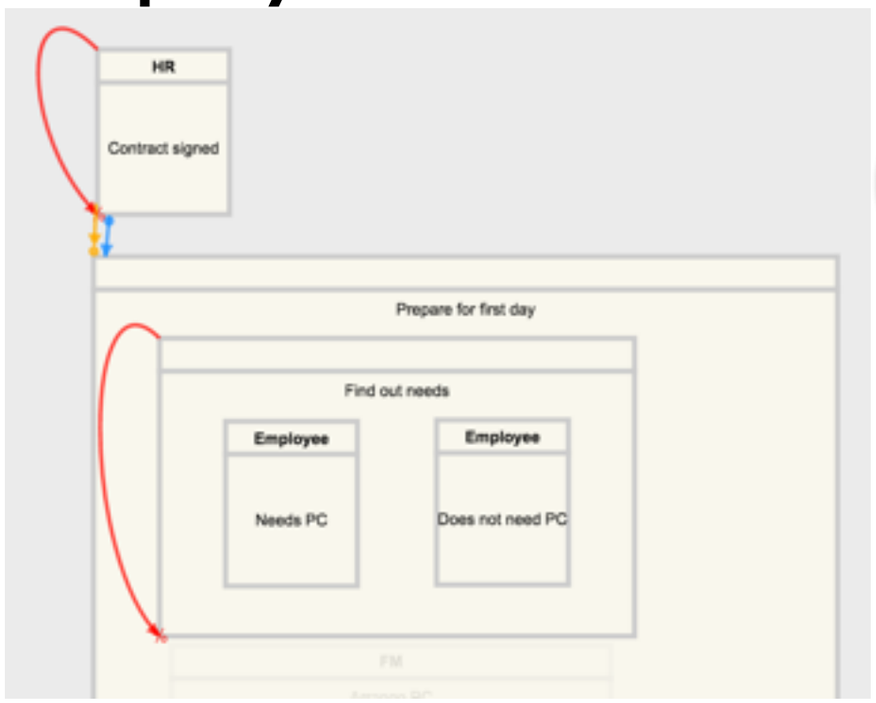
Papers: [SEFM2011,FHIES2011]



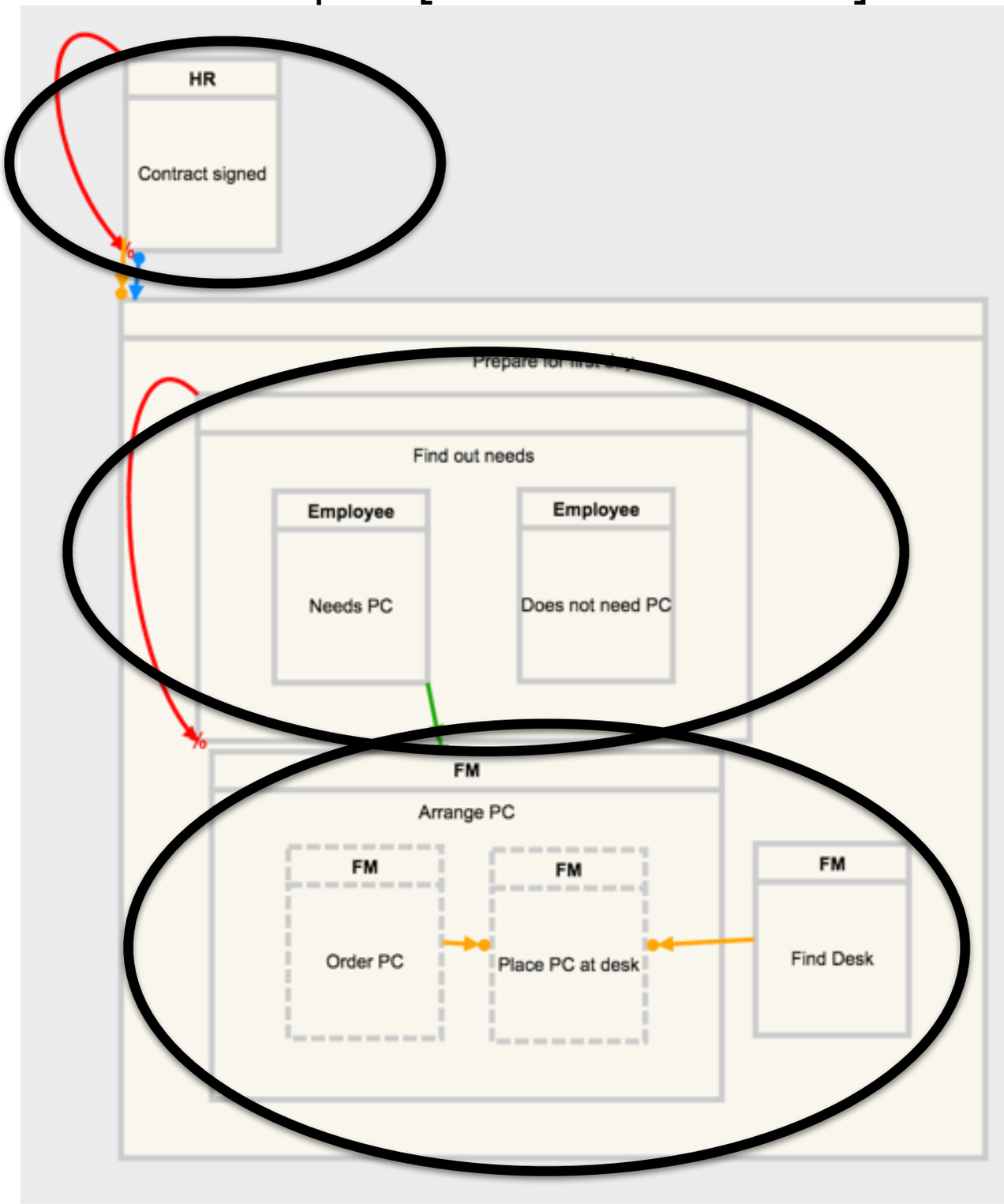
Thomas T. Hildebrandt (hilde@itu.dk)

HR

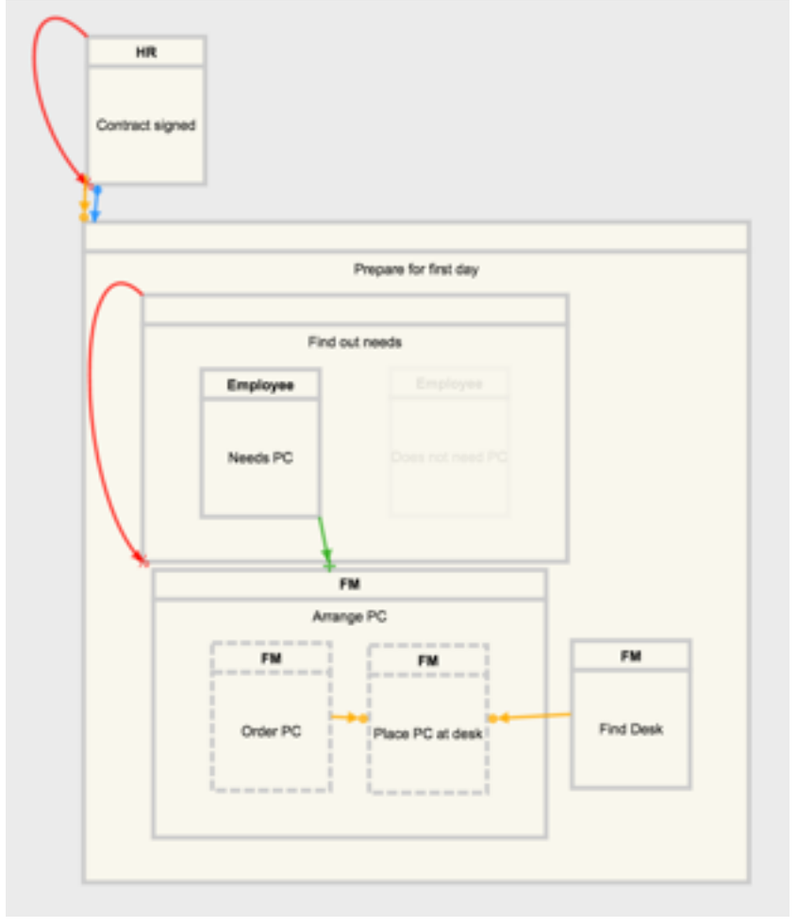
Employee



Papers: [SEFM2011,FHIES2011]



FM



Thomas T. Hildebrandt (hilde@itu.dk)

IT UNIVERSITY OF COPENHAGEN

Thomas T. Hildebrandt (hilde@itu.dk)

IT UNIVERSITY OF COPENHAGEN

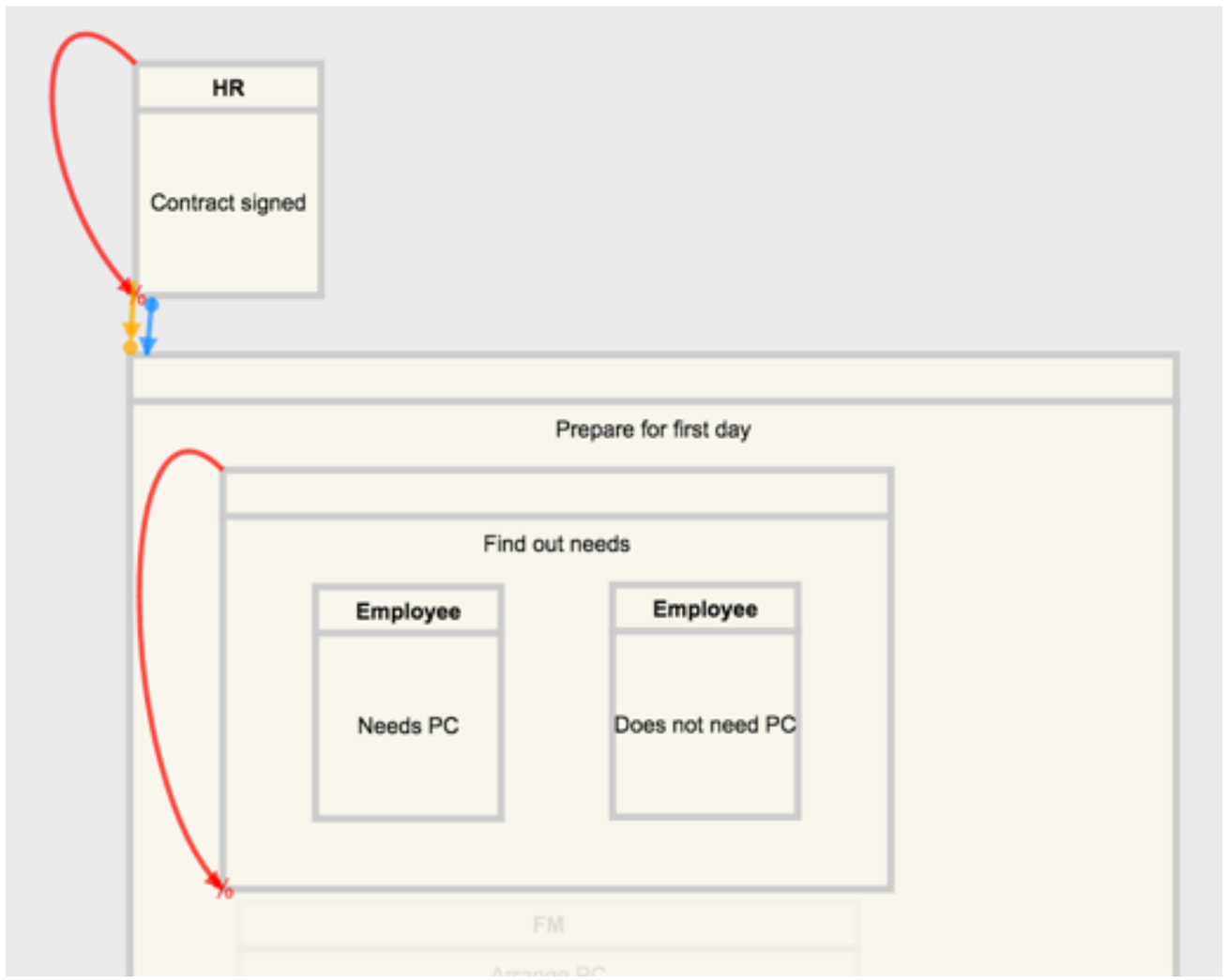
HR

Thomas T. Hildebrandt (hilde@itu.dk)

IT UNIVERSITY OF COPENHAGEN

HR

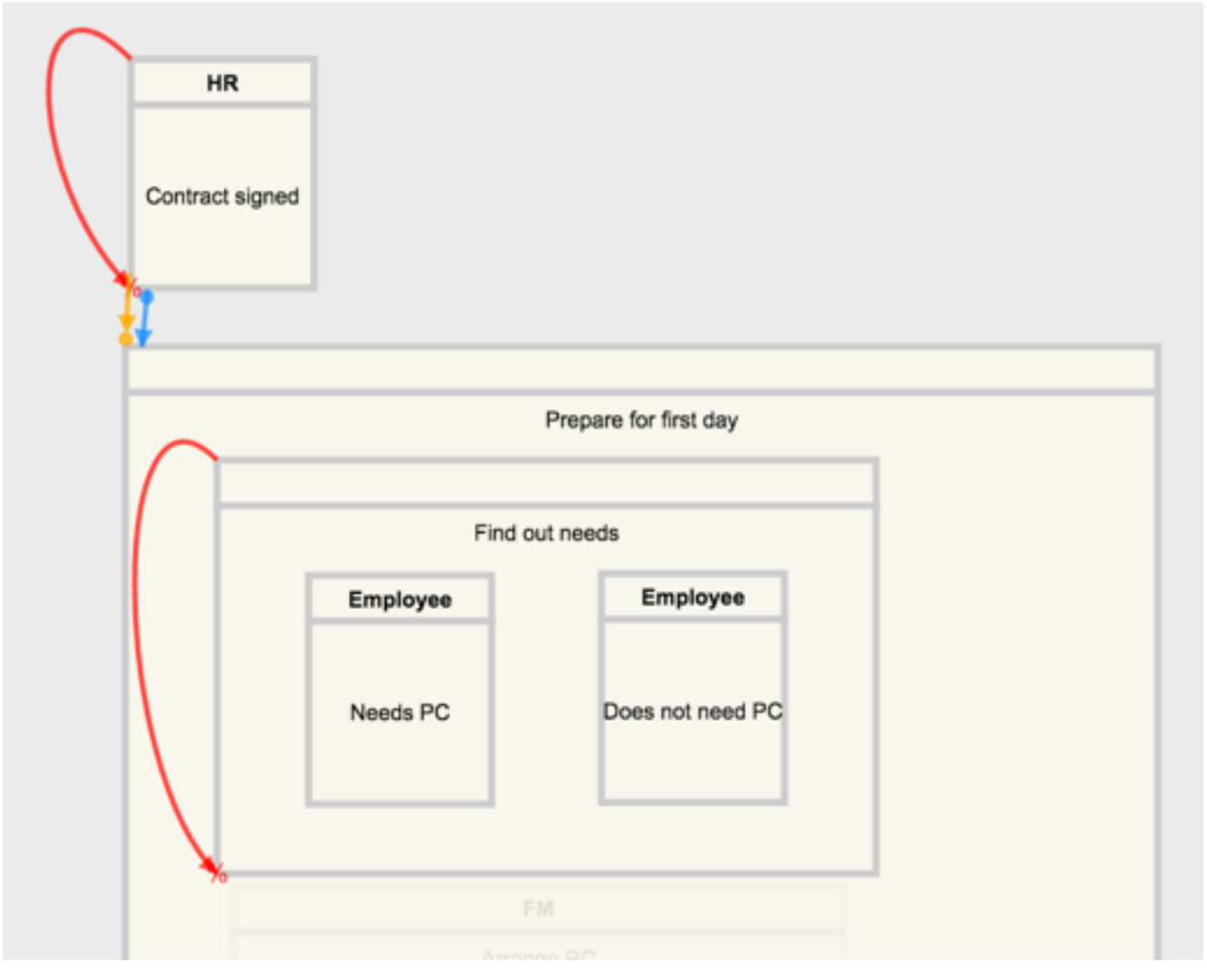
Employee



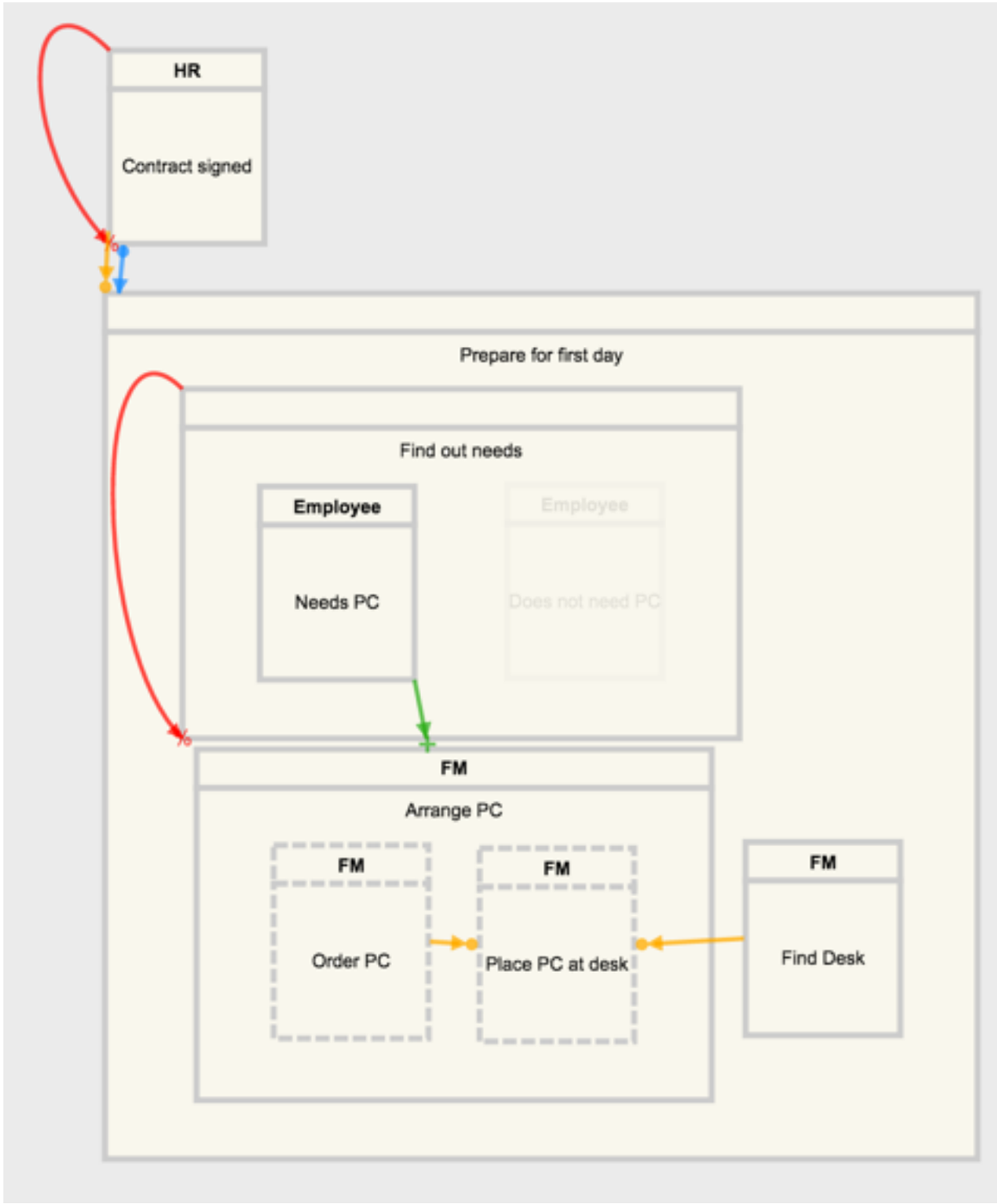
Thomas T. Hildebrandt (hilde@itu.dk)

HR

Employee



FM

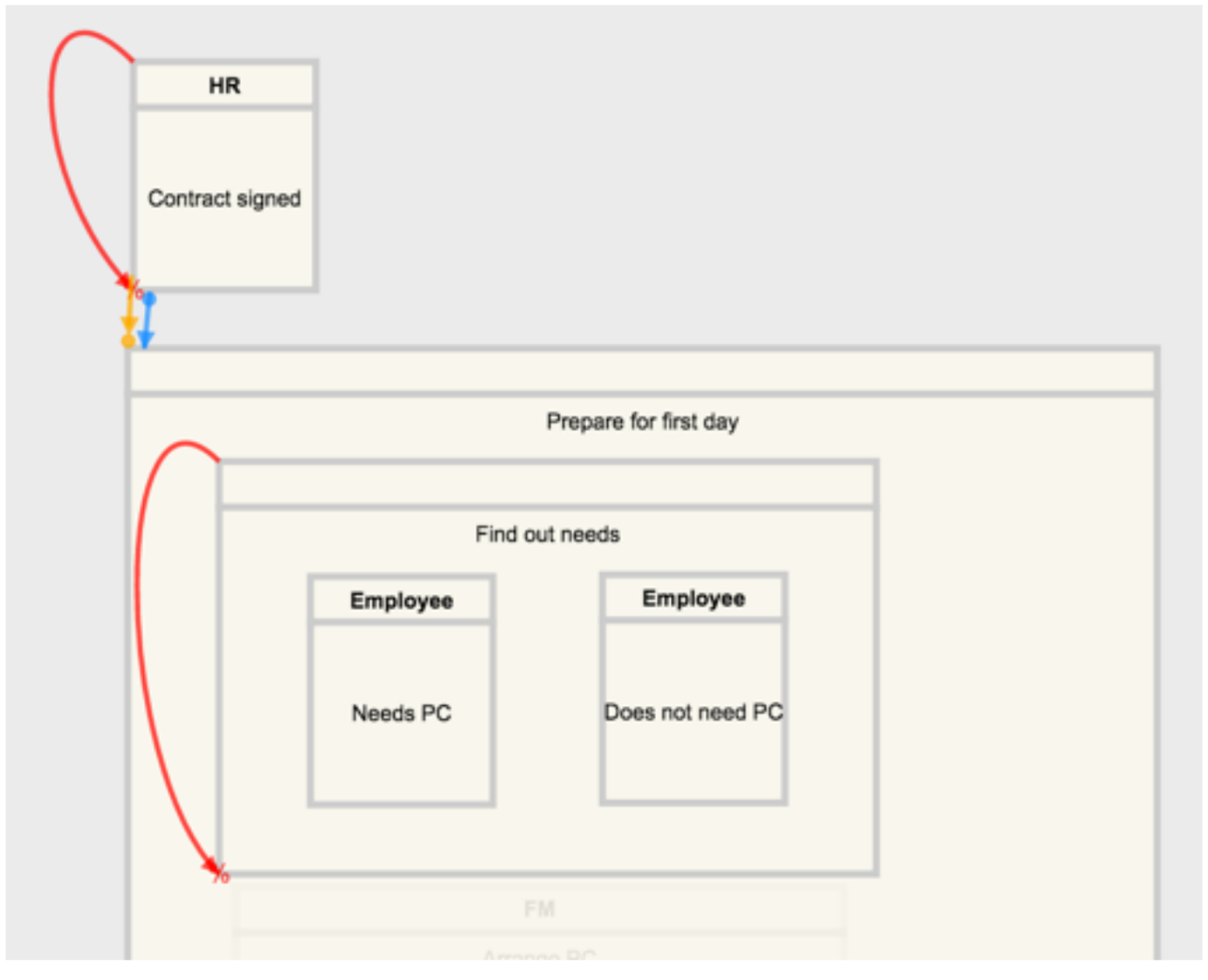


Thomas T. Hildebrandt (hilde@itu.dk)

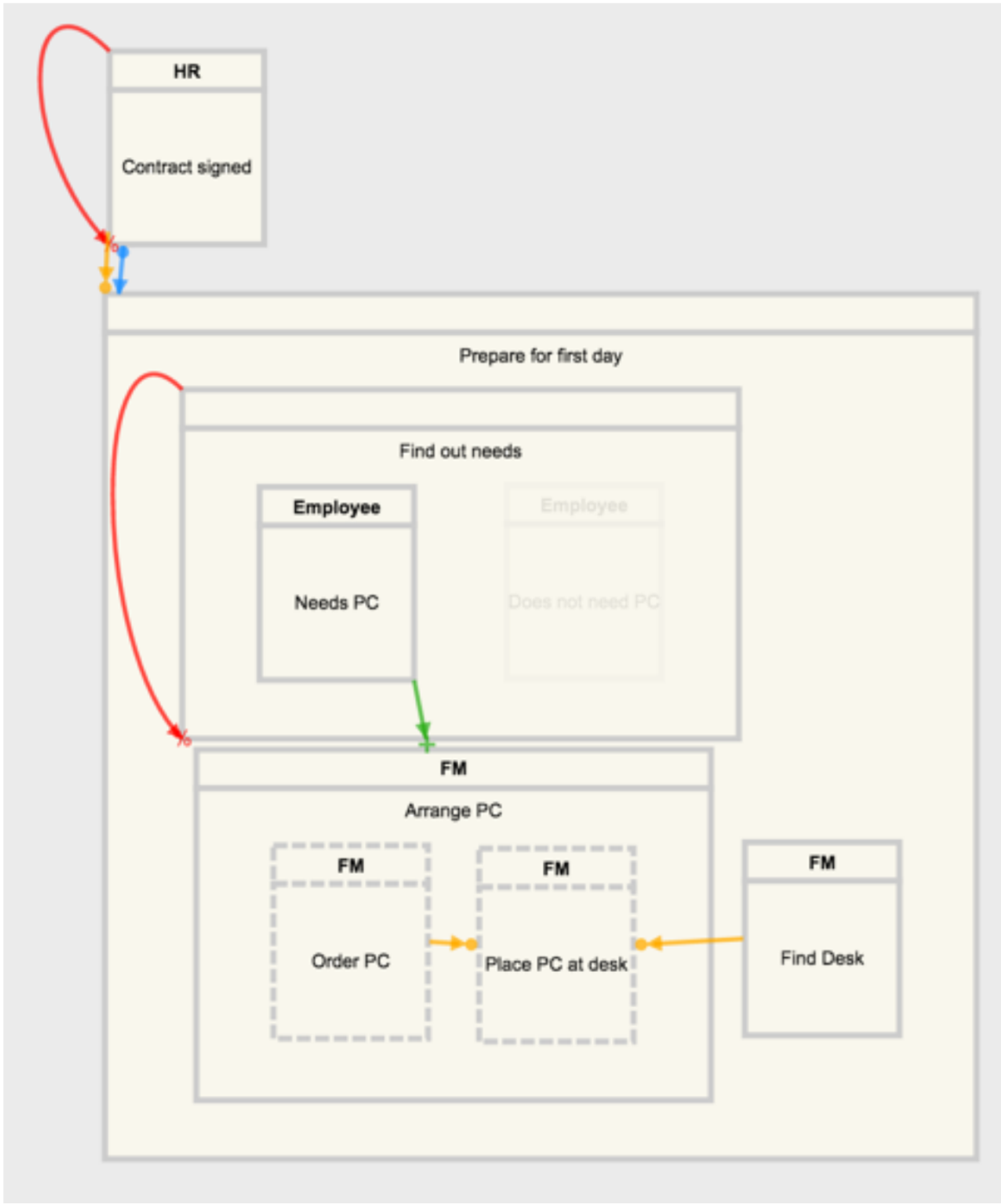
SEFM2015: Each role subscribes to remote events

HR

Employee

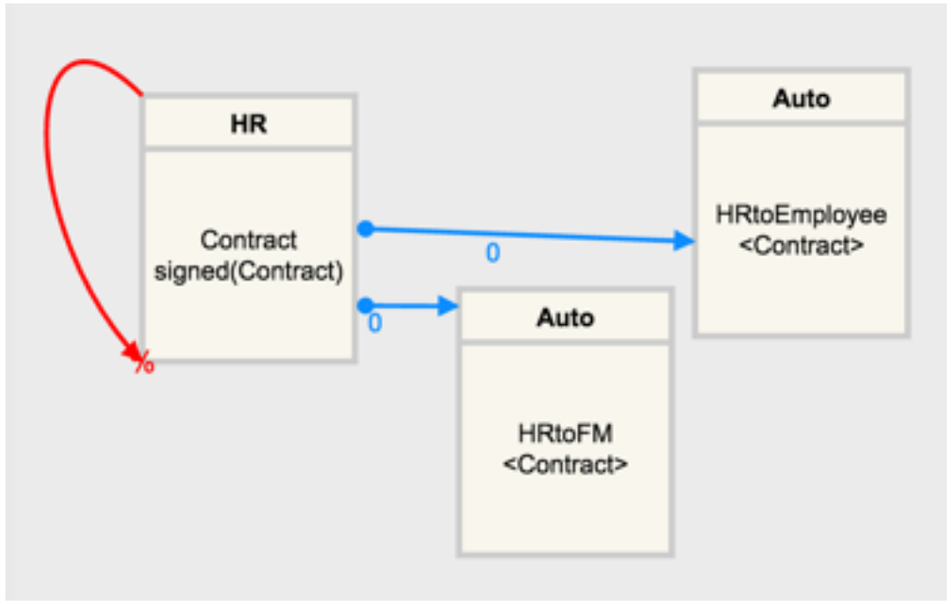


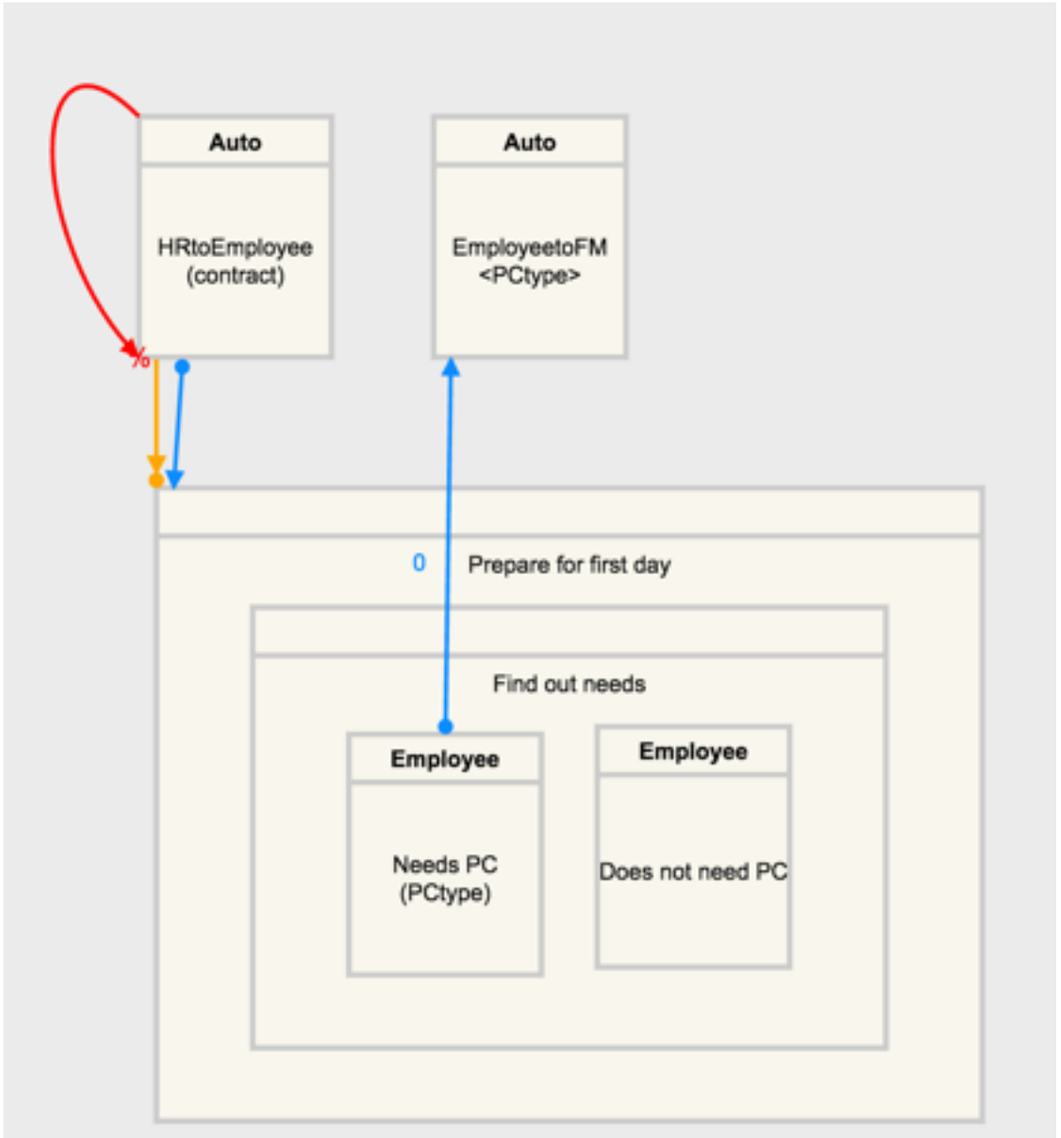
FM

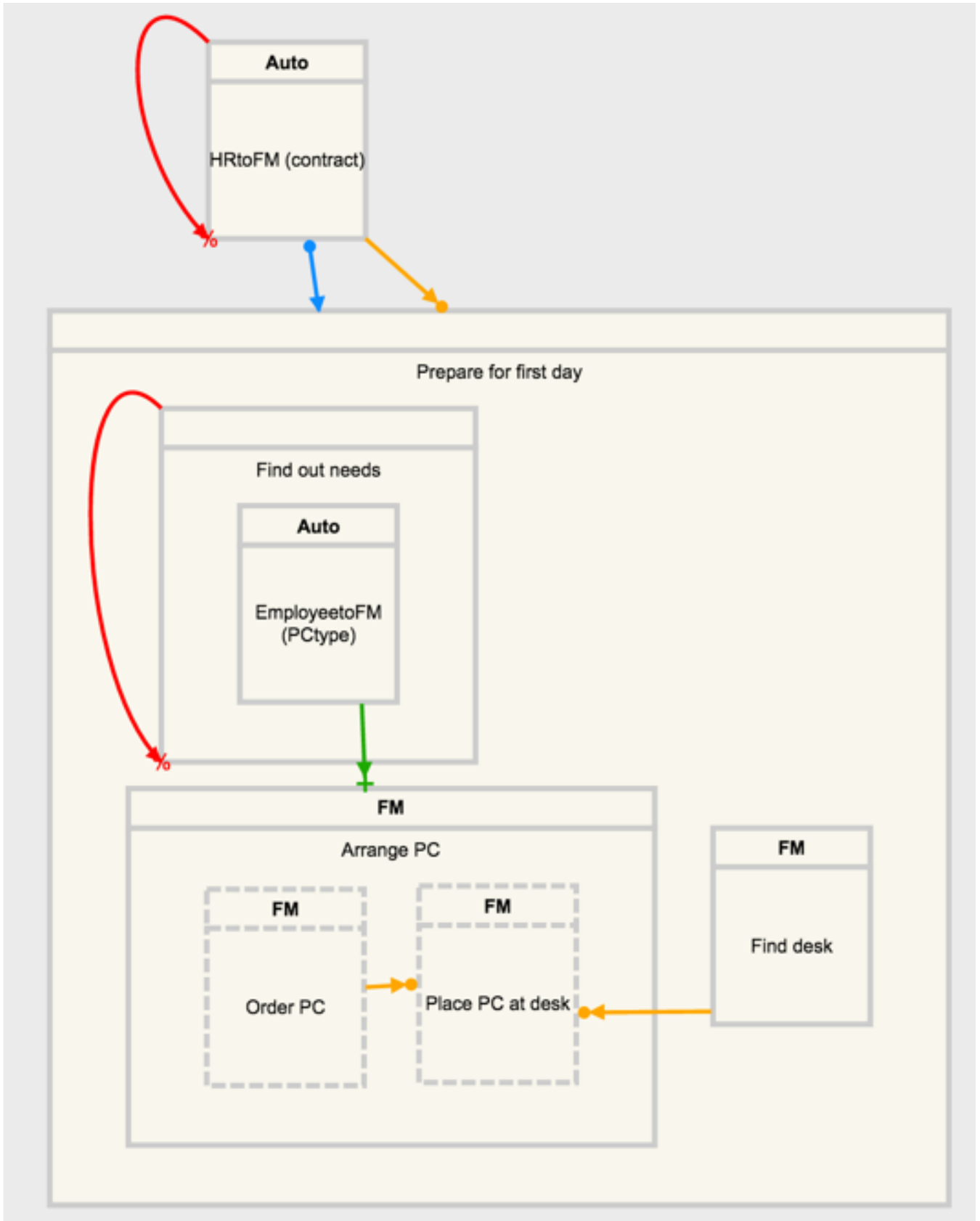


Thomas T. Hildebrandt (hilde@itu.dk)

IT UNIVERSITY OF COPENHAGEN

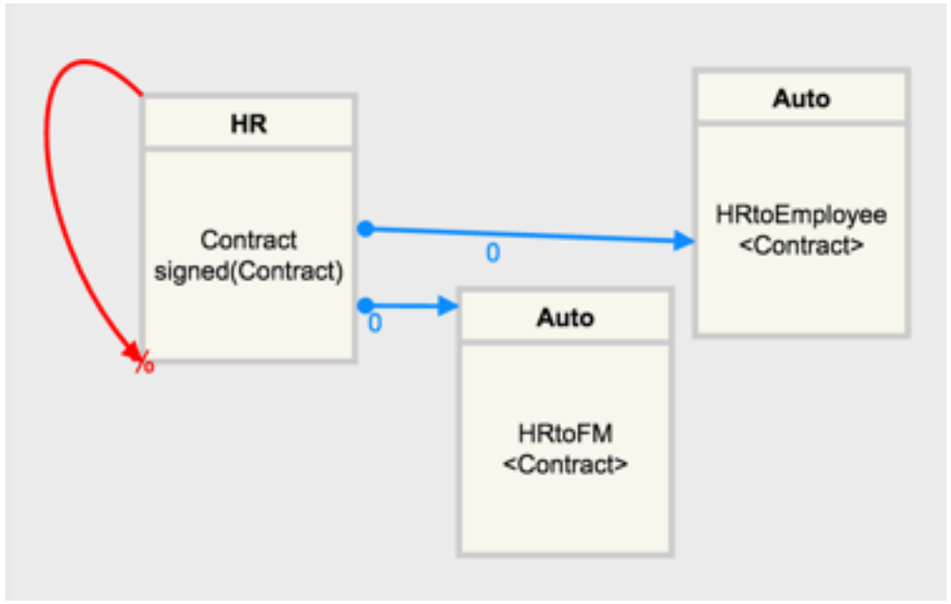


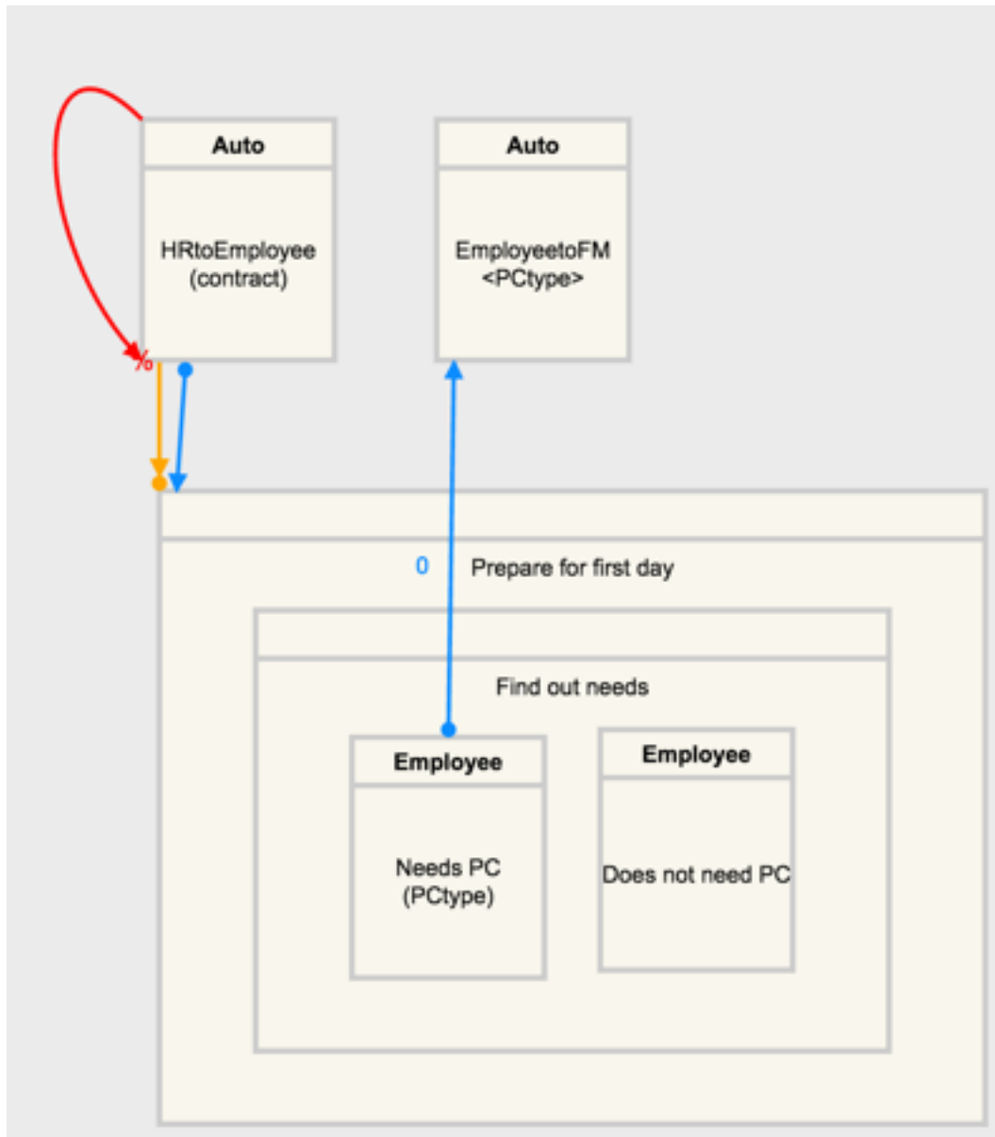


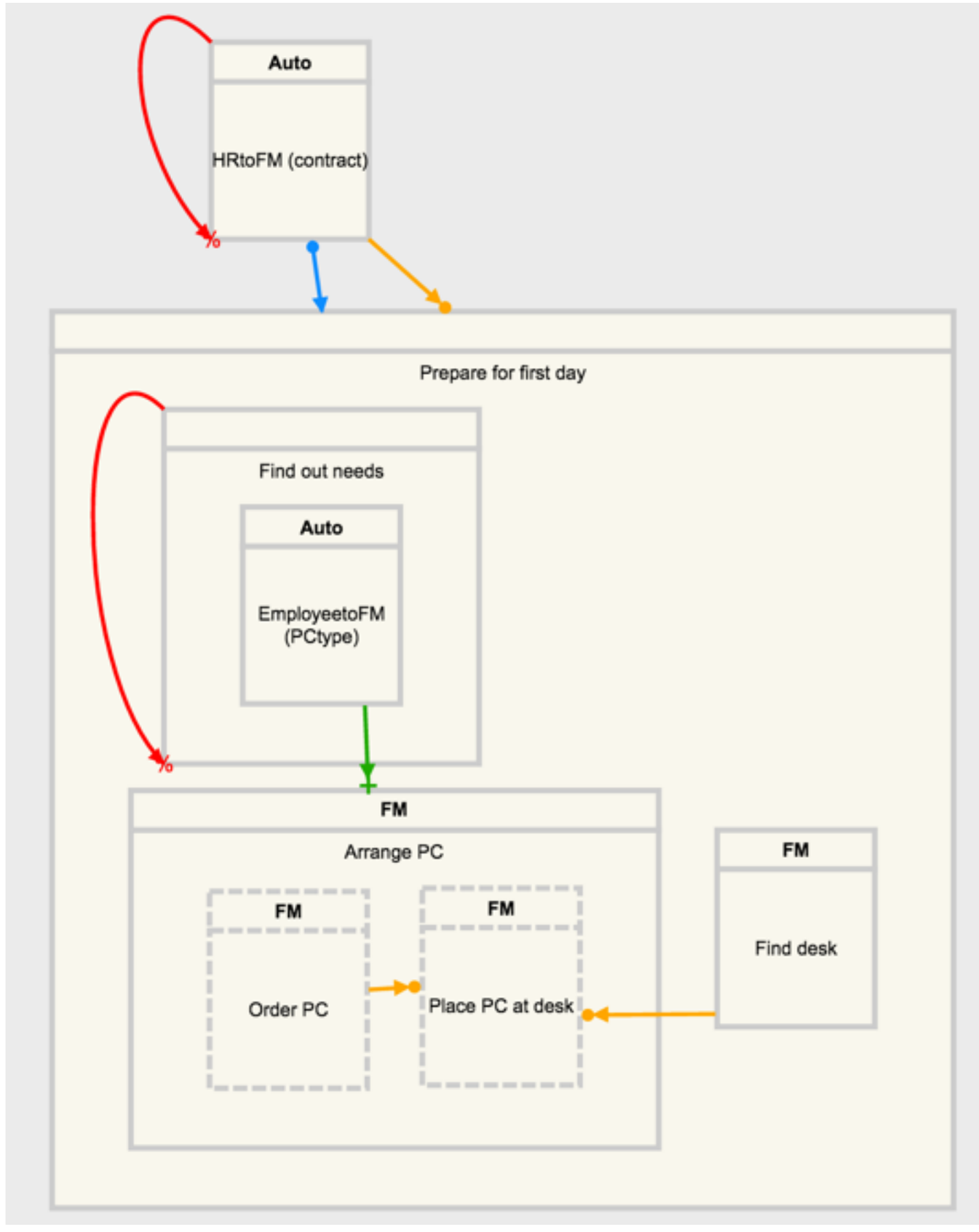


Thomas T. Hildebrandt (hilde@itu.dk)

Explicit communication events & data







Thomas T. Hildebrandt (hilde@itu.dk)

Conclusion

Conclusion

- DCR Graphs provide an adaptable, declarative alternative to imperative control, safety and liveness conditions - *replacing choice, sequencing and loop*ing with include/exclude, conditions and responses

Conclusion

- DCR Graphs provide an adaptable, declarative alternative to imperative control, safety and liveness conditions - *replacing choice, sequencing and loop*ing with include/exclude, conditions and responses
- Extensions with sub processes, time, data, communication

Conclusion

- DCR Graphs provide an adaptable, declarative alternative to imperative control, safety and liveness conditions - *replacing choice, sequencing and loop*ing with include/exclude, conditions and responses
- Extensions with sub processes, time, data, communication
- Support for execution, safe distribution, monitoring & proactive enforcement

Conclusion

- DCR Graphs provide an adaptable, declarative alternative to imperative control, safety and liveness conditions - *replacing choice, sequencing and loop*ing with include/exclude, conditions and responses
- Extensions with sub processes, time, data, communication
- Support for execution, safe distribution, monitoring & proactive enforcement

Data example

Input events:

Hours? HourlyRate? TaxRate?

Computation events:

$[= \mathsf{HourlyRate} * \mathsf{Hours}]\mathsf{Wage}$ $[= \mathsf{Wage} * (1 - \mathsf{TaxRate})]\mathsf{WageAfterTax}$

Data example

Input events:

Hours? HourlyRate? TaxRate?

Computation events:

$[= \text{HourlyRate} * \text{Hours}]\text{Wage}$ $[= \text{Wage} * (1 - \text{TaxRate})]\text{WageAfterTax}$

Dynamics/reactions/behaviour:

$\text{Hours} \bullet\xrightarrow{0} \text{Wage} \mid \text{HourlyRate} \bullet\xrightarrow{0} \text{Wage} \mid \text{TaxRate} \bullet\xrightarrow{0} \text{WageAfterTax} \mid \text{Wage} \bullet\xrightarrow{0} \text{WageAfterTax}$

Data example

Input events:

Hours? HourlyRate? TaxRate?

Computation events:

$[= \text{HourlyRate} * \text{Hours}]\text{Wage}$ $[= \text{Wage} * (1 - \text{TaxRate})]\text{WageAfterTax}$

Dynamics/reactions/behaviour:

$\text{Hours} \bullet\xrightarrow{0} \text{Wage} \mid \text{HourlyRate} \bullet\xrightarrow{0} \text{Wage} \mid \text{TaxRate} \bullet\xrightarrow{0} \text{WageAfterTax} \mid \text{Wage} \bullet\xrightarrow{0} \text{WageAfterTax}$

$\text{Hours} \bullet\xleftarrow{0} \text{HourlyRate} \mid \text{Hours} \bullet\xleftarrow{0} \text{TaxRate}$

Communication & Obligations

$$\mathsf{RequestPayment} \bullet \xrightarrow{0;\mathsf{WageAfterTax}\neq\bot} \mathsf{Finance} \mid \mathsf{Finance}\langle\mathsf{WageAfterTax}\rangle \mid$$

$$\mathsf{Finance} \rightarrow\!\!\!\!\% \ \mathsf{RequestPayment} \mid \mathsf{Answer} \rightarrow\!\!\!\!+ \ \mathsf{RequestPayment} \mid \mathsf{Answer?} \mid \mathsf{RequestPayment?}$$

$$\mathsf{Finance?ToPay} \mid \mathsf{Decision?} \mid \mathsf{Answer}\langle\mathsf{Decision}\rangle\mathsf{SendAnswer} \mid \mathsf{ToPay} \bullet\!\!\xrightarrow{\omega} \mathsf{Decision}$$

$$\mathsf{Decision} \bullet\!\!\xrightarrow{20} \mathsf{SendAnswer} \mid \mathsf{SendAnswer} \bullet\!\!\xleftarrow{10} \mathsf{Decision}$$