Technologies for Flexible Cross-organizational Case Management Systems

Tijs Slaats

Joint work with
Thomas Hildebrandt and Raghava Rao Mukkamala

eXformatics

IT University of Copenhagen

Overview

- Project Background
- Project Goals
- DCR Graphs
  - Informal introduction
  - Formal semantics
  - Distribution
  - Data and Time
  - Tools
- Future work

Industrial PhD

IT University of Copenhagen

- Enrolled as PhD Student
- 50% of time (plus 6 months abroad)
- Research, classes, teaching

eXformatics

- Employer (pays salery, travel expenses etc)
- 50% of time
- Tool development, case studies, knowledge dissemination (internal and external presentations, etc)

Danish Government

- Subsidizes project, covers:
- All expenses university
- Approx. half of expenses company

IT University of Copenhagen

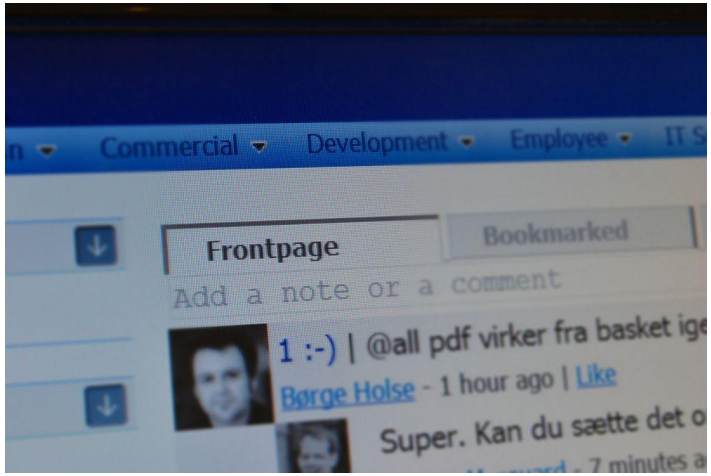
- Founded in 1999
- Small independent university (1,500 students)
- Focused on different areas of IT:
  - Communication
  - Business
  - Software Development
  - Games

eXformatics

IT University of Copenhagen

Exformatics

- Danish company, with a branch in Sweden
- Founded in 2003
- Approx. 20 employees
- Develops Electronic Case- and Document-handling system



eXformatics

IT University of Copenhagen

Project Goals

Technologies for Flexible Cross-organizational Case Management Systems

- *Case Management* originates from social work and health care, the same concept has since popped up in many other fields under a variety of new names such as *case handling* and *adaptive case management.*

Case Management

- Centered on the concept of a ***case***, a process with (usually) the following properties:
  - Aims to solve the needs of a *client* (customer).
  - Goals are *customized* to individual clients and cases.
  - There is a main responsible: the *case worker*.
  - Requires *professional knowledge* and *decision-making*.
  - Involves many participants across multiple divisions and/or organizations

Case Management

- Examples:
  - legal cases in a lawyers firm,
  - claims in an insurance company,
  - hiring of an employee in a HR department,
  - treatment of a patient in a hospital.

Project Goals

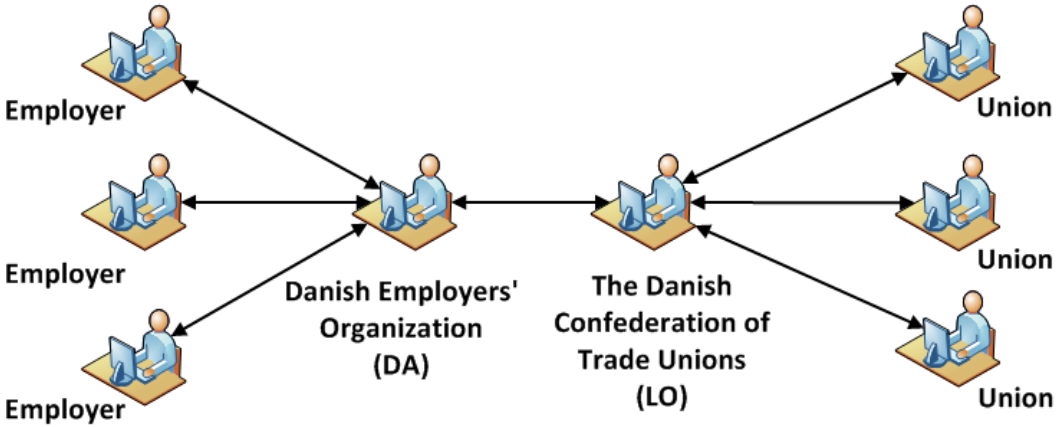
Technologies for Flexible Cross-organizational Case Management Systems

- *Caseworkers* are *knowledge-workers*: they require flexibility to make their own decisions.
- *Declarative* workflow languages are better at offering this flexibility then *imperative* workflow languages.
- Our declarative model: **DCR Graphs**.

Project Goals

Technologies for Flexible Cross-organizational Case Management Systems

- Case management often involves multiple (sub-) organizations.
- Example: Employer-Employee conflicts in Denmark.



eXformatics

IT University of Copenhagen

DCR Graphs

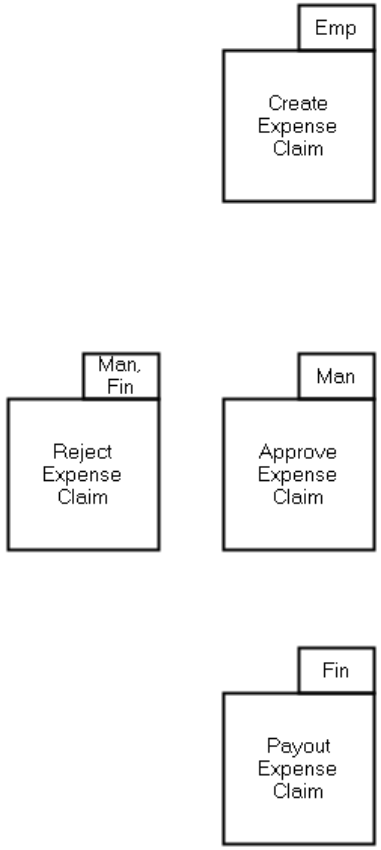
- A declarative workflow language, generalization of *event structures*.
- Consists of *events* and *constraints* between events
- Unconstrained events can happen at any time
- State represented as a *marking* consisting of *executed*, *pending* and *included* events

DCR Graphs

- DCR Graphs are not the first declarative workflow model, but adds:
  - Only 4 basic relations, yet equal to ω-regular languages in expressiveness.
  - Semantics based on transformation of markings: makes execution of DCR Graphs easy to understand and represent graphically.

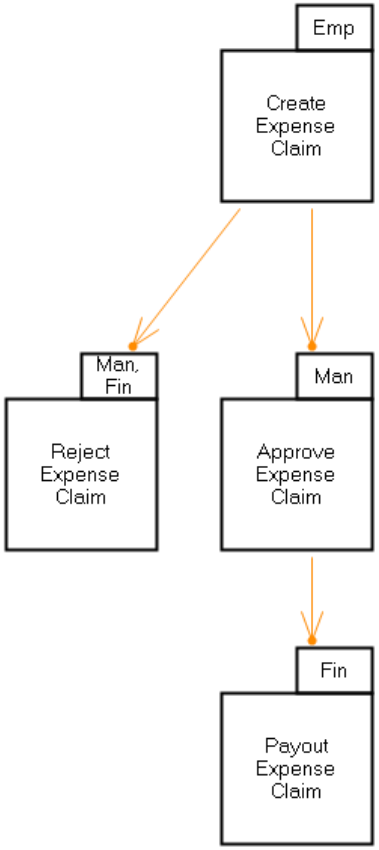
DCR Graphs: Events and Roles

- 4 main activities:
  - Create Expense Claim
  - Approve Expense Claim
  - Reject Expense Claim
  - Payout Expense Claim
- Activities limited to specific roles:
  - Employee
  - Manager
  - Finance Department



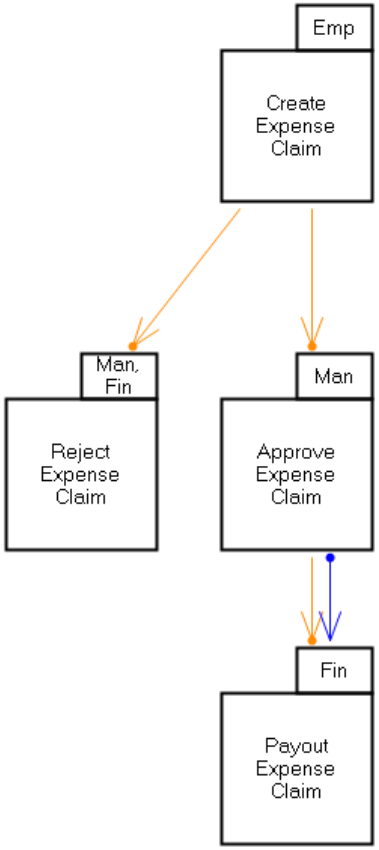
DCR Graphs: Conditions

- A claim should be created before it can be approved or rejected.
- A claim should be approved before it can be paid out.



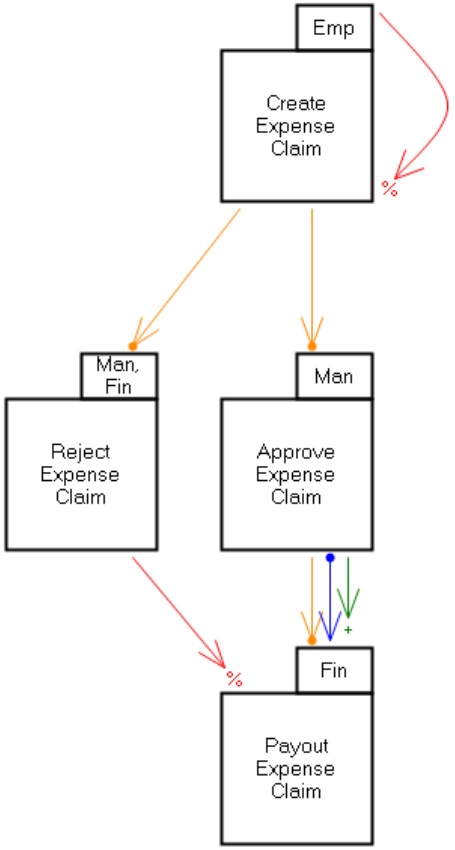
DCR Graphs: Responses

- Once a claim has been approved, it should *eventually* be paid out.



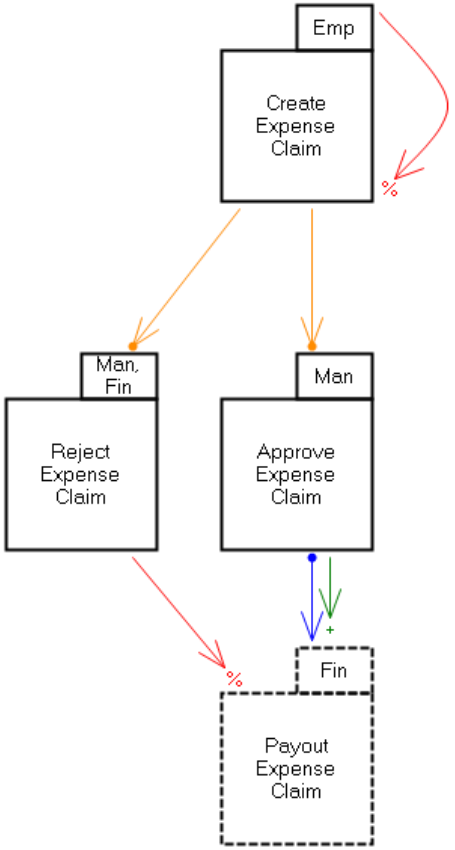
DCR Graphs: Exclusions and Inclusions

- A claim should only be created once. (Every run of the workflow handles a single claim.)
- Once a claim has been rejected, it should not be paid out, unless it is approved again at some later point in time.



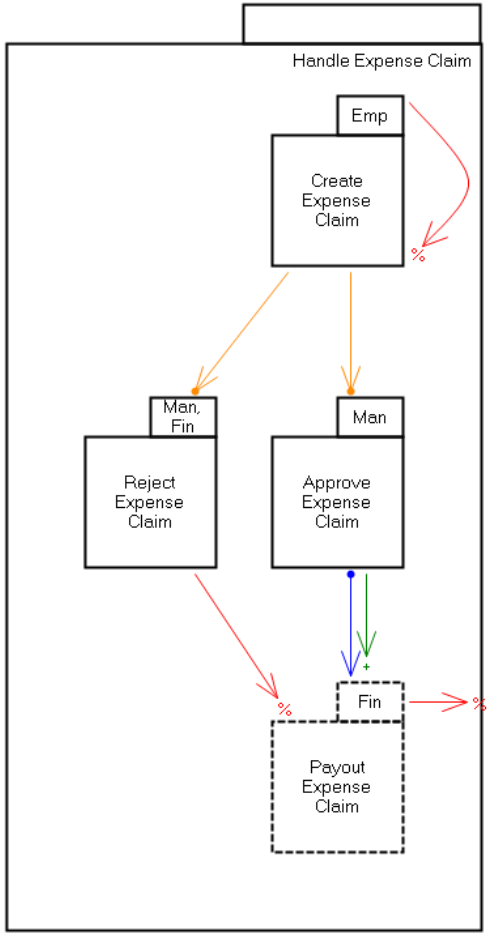
DCR Graphs: Initial Marking

- We can remove the condition from *Approve* to *Payout* if we exclude *Payout* initially. (*Approve* still needs to be executed before it, because it is the only event that includes it).



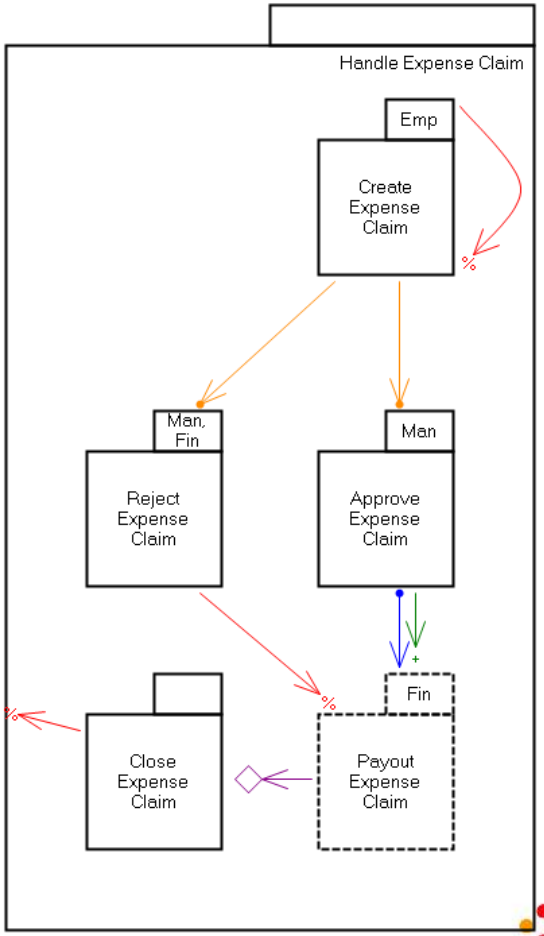
DCR Graphs: Nesting

- Payout should end the process and therefor exclude everything.
- Nesting as a shorthand for having an exclude relation to all four events.



DCR Graphs: Milestone

- Case closed manually, can happen at any time, unless we still need to pay out the claim.
- We will see later that the milestone relation captures the acceptance condition for finite runs.



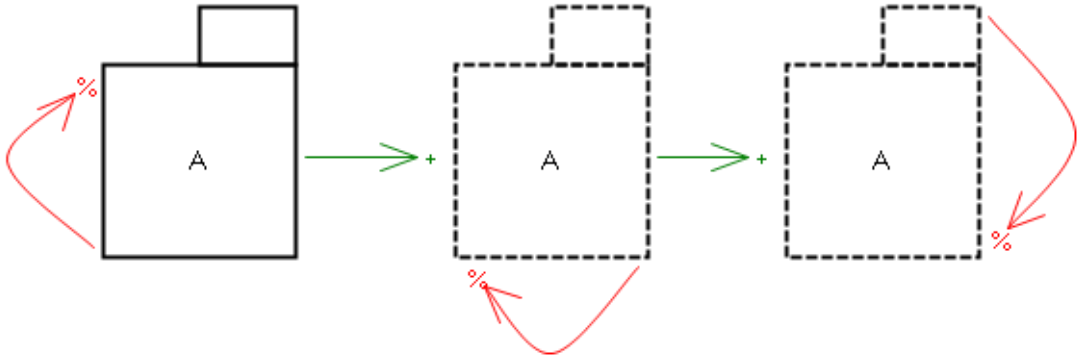
DCR Graphs: Formal Definition

A Nested DCR Graph G is a tuple $G = (\mathsf{E}, \rhd, \mathsf{M}, \rightarrow\bullet, \bullet\rightarrow, \rightarrow\diamond, \rightarrow+, \rightarrow\%, \mathsf{L}, l)$, *where*

(i) E *is the set of* events
(ii) $\rhd : \mathsf{E} \rightharpoonup \mathsf{E}$ *is a partial function mapping an event to its super-event*
(iii) $\mathsf{atoms}(\mathsf{E})$ *is the set of atomic events that do not have any sub-events*
(iv) $\mathsf{M} \in \mathcal{P}(\mathsf{atoms}(\mathsf{E})) \times \mathcal{P}(\mathsf{atoms}(\mathsf{E})) \times \mathcal{P}(\mathsf{atoms}(\mathsf{E}))$ *is the* initial marking
(v) $\rightarrow\bullet \subseteq \mathsf{E} \times \mathsf{E}$ *is the* condition *relation*
(vi) $\bullet\rightarrow \subseteq \mathsf{E} \times \mathsf{E}$ *is the* response *relation*
(vii) $\rightarrow\diamond \subseteq \mathsf{E} \times \mathsf{E}$ *is the* milestone *relation*
(viii) $\rightarrow+ \subseteq \mathsf{E} \times \mathsf{E}$ *is the* dynamic inclusion *relation*
(ix) $\rightarrow\% \subseteq \mathsf{E} \times \mathsf{E}$ *is the* dynamic exclusion *relation*
(x) L *is the set of* labels *(e.g. activities + roles)*
(xi) $l : \mathsf{E} \rightarrow \mathcal{P}(\mathsf{L})$ *is a labeling function mapping events to sets of labels.*

DCR Graphs: Labels

- Allow multiple events to be mapped to the same label.
- For example allows to express:



- Also introduces non-determinism if only labels are observed (two events with the same label may be enabled at the same time).

DCR Graphs: Formal Semantics

- Nested DCR Graph is flattened into a regular DCR Graph (relations go from/to all sub-events and super-events are discarded).

- An event is *enabled* if:

$$M \vdash_G e, \textit{if } e \in \mathsf{In} \textit{ and } (\mathsf{In} \cap \rightarrow\bullet\, e) \subseteq \mathsf{Ex} \textit{ and } (\mathsf{In} \cap \rightarrow\diamond\, e) \subseteq \mathsf{E} \setminus \mathsf{Re}$$

Event is included

All *included* condition events have been executed

None of the *included* milestone events is a pending response

DCR Graphs: Formal Semantics

- Result of executing an event:

$$(\mathsf{Ex}, \mathsf{Re}, \mathsf{In}) \oplus_G e = \big(\mathsf{Ex} \cup \{e\}, (\mathsf{Re} \setminus \{e\}) \cup e \bullet\!\!\rightarrow, (\mathsf{In} \setminus e \rightarrow \%) \cup e \rightarrow +\big)$$

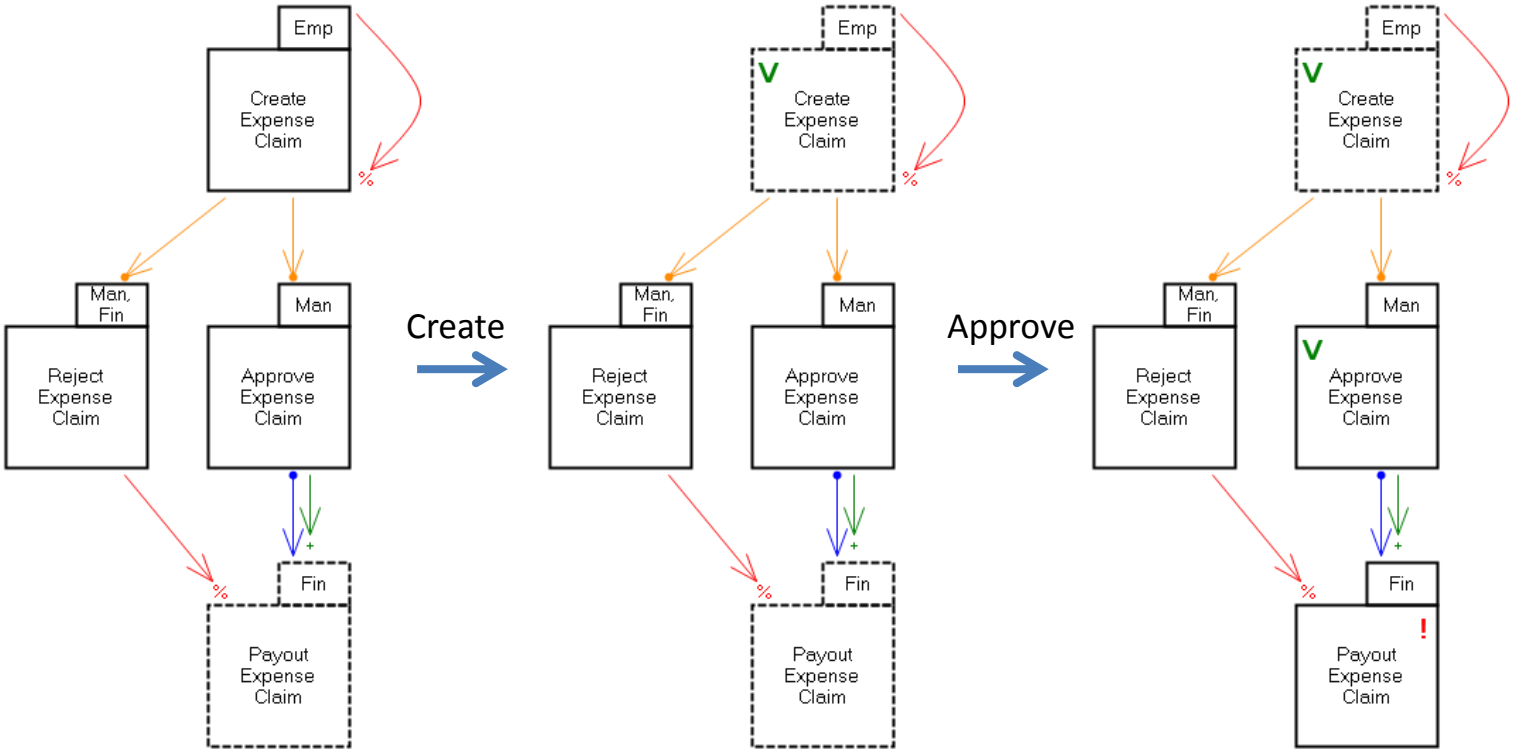
Event *e* is added to the set of executed events

Event *e* is removed from the set of pending responses. Afterwards all events that are a response to *e* are added to the set of pending responses.

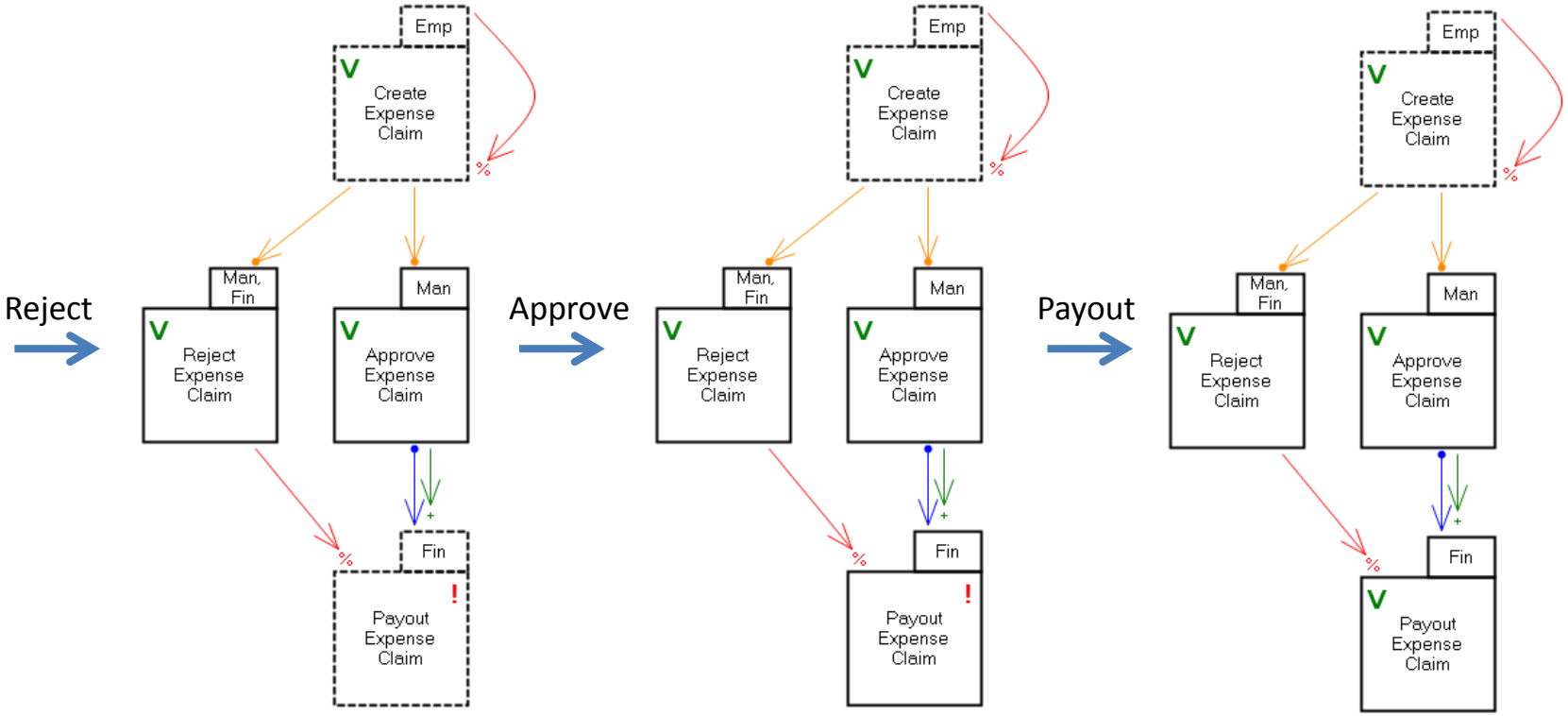
All events that are excluded by *e* are removed from the set of included events. Afterwards all events that are included by *e* are added to the set of included events.

- Note that:
  - If an event is a response to itself, it will remain in the set of pending responses after execution.
  - If an event e both includes and excludes an event e', e' will always be included after execution of e.

DCR Graphs: Example Run



DCR Graphs: Example Run

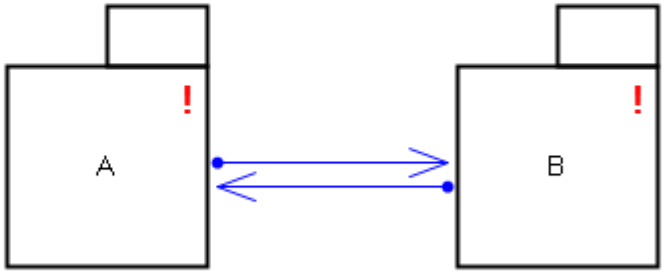


eXformatics

IT University of Copenhagen

DCR Graphs: Accepting runs

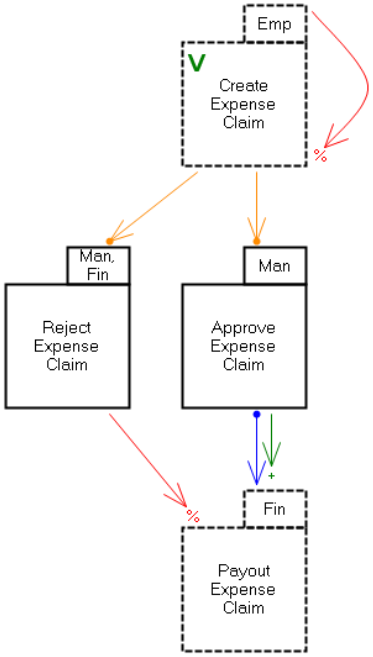
- A run is accepting if every pending event is eventually executed or excluded.
- Finite runs: Any marking where none of the included events are also pending is accepting.
- Infinite runs:

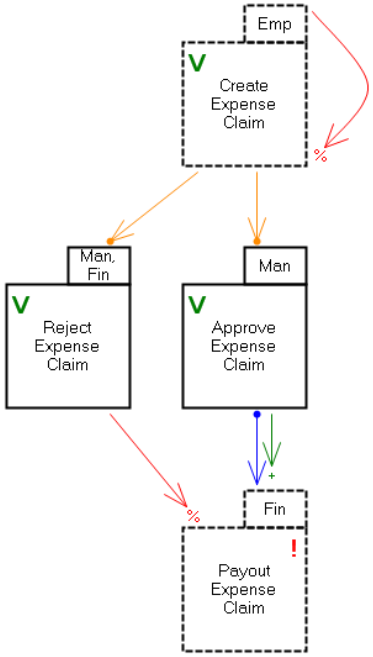


A^{ω} is not accepting, but $(AB)^{\omega}$ is.

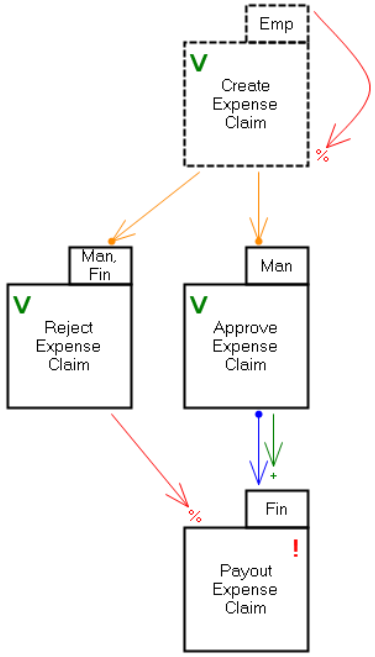
DCR Graphs: Accepting markings

Accepting:





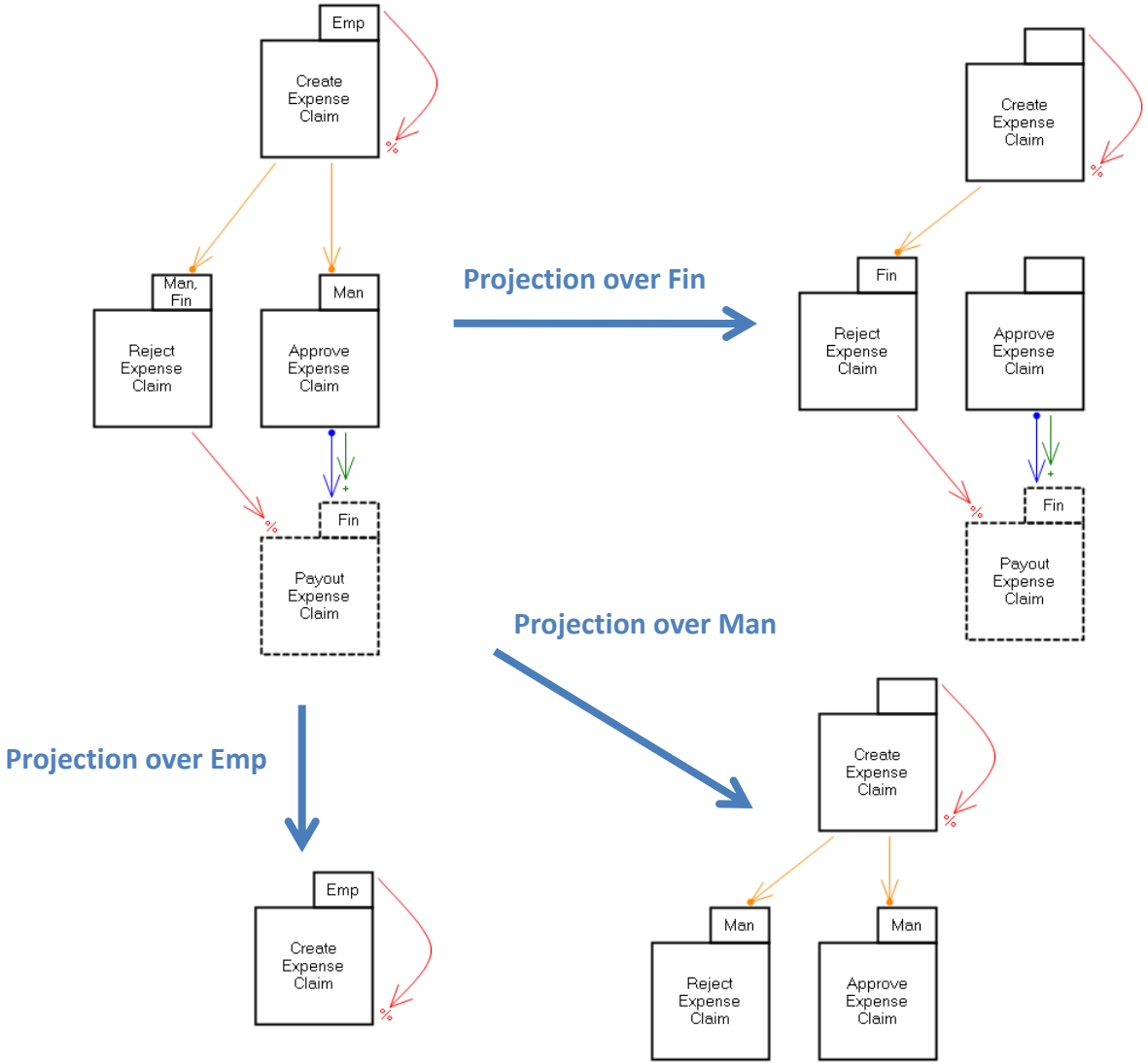
Not Accepting:



eXformatics

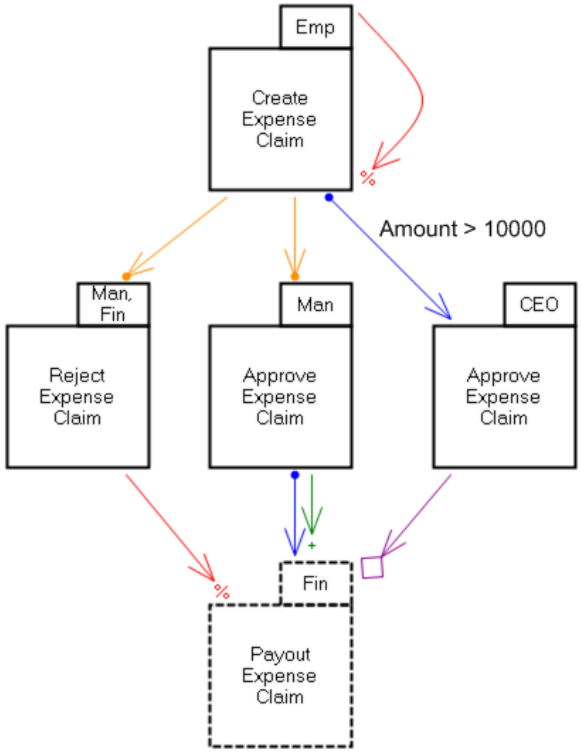
IT University of Copenhagen

DCR Graphs: Distribution



DCR Graphs: Global Data

- Global data store
- Events and relations can be guarded by expressions on variables in the data store.
- Example: If the amount of an expense claim is larger then 10000 euros, the CEO needs to approve it as well.

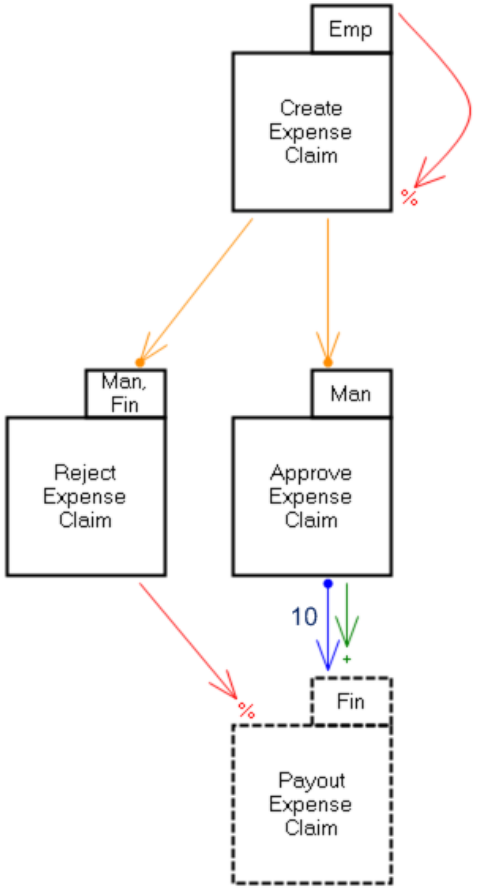


DCR Graphs: Parameterized Events

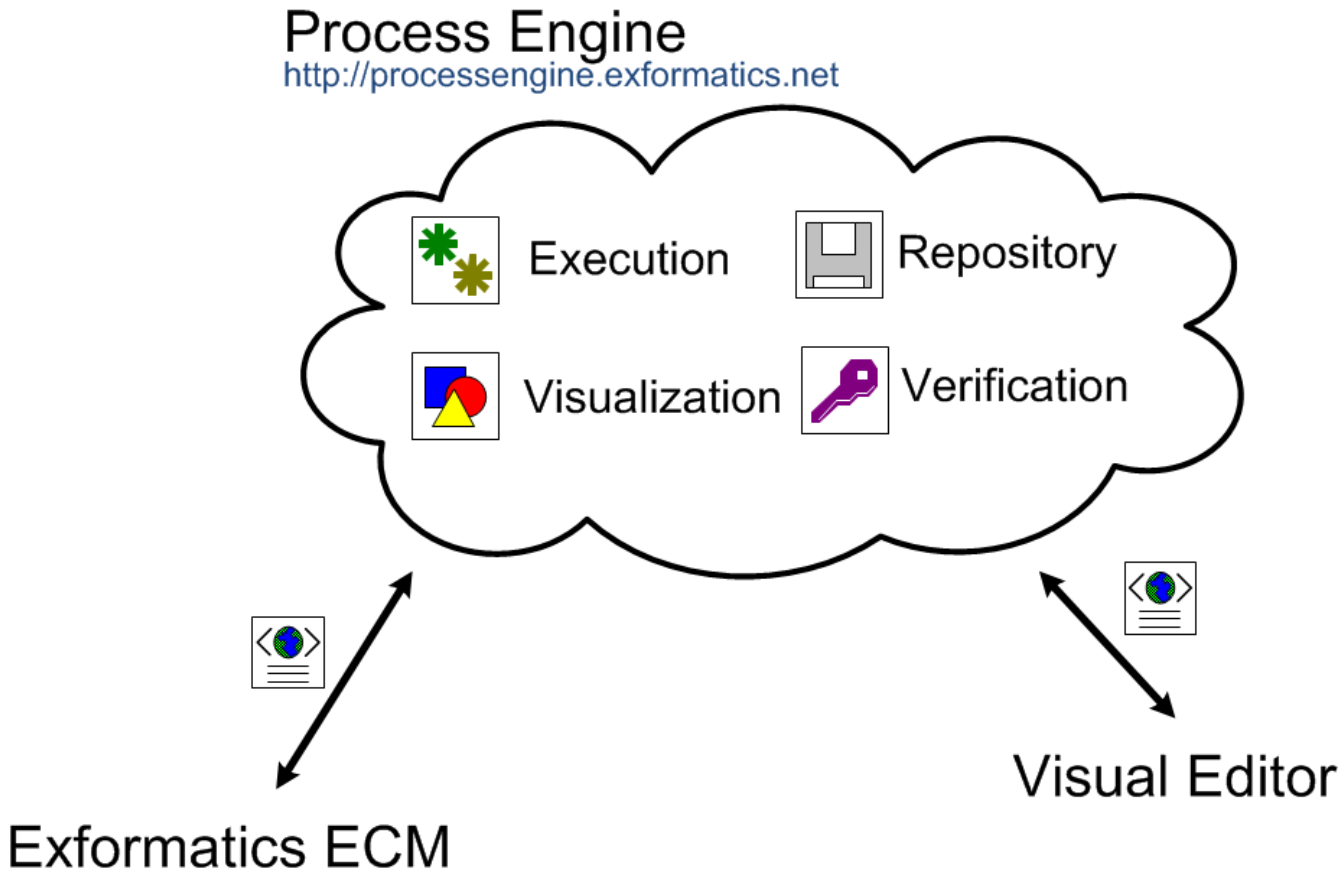
- Events are parameterized, for example: *Create Expense Claim (string Name, int Amount)*
- Relations can be parameterized, for example the *Create Expense Claim* event above can be a condition for an *Approve Expense Claim* event with the same *Name* and *Amount*.

DCR Graphs: Time

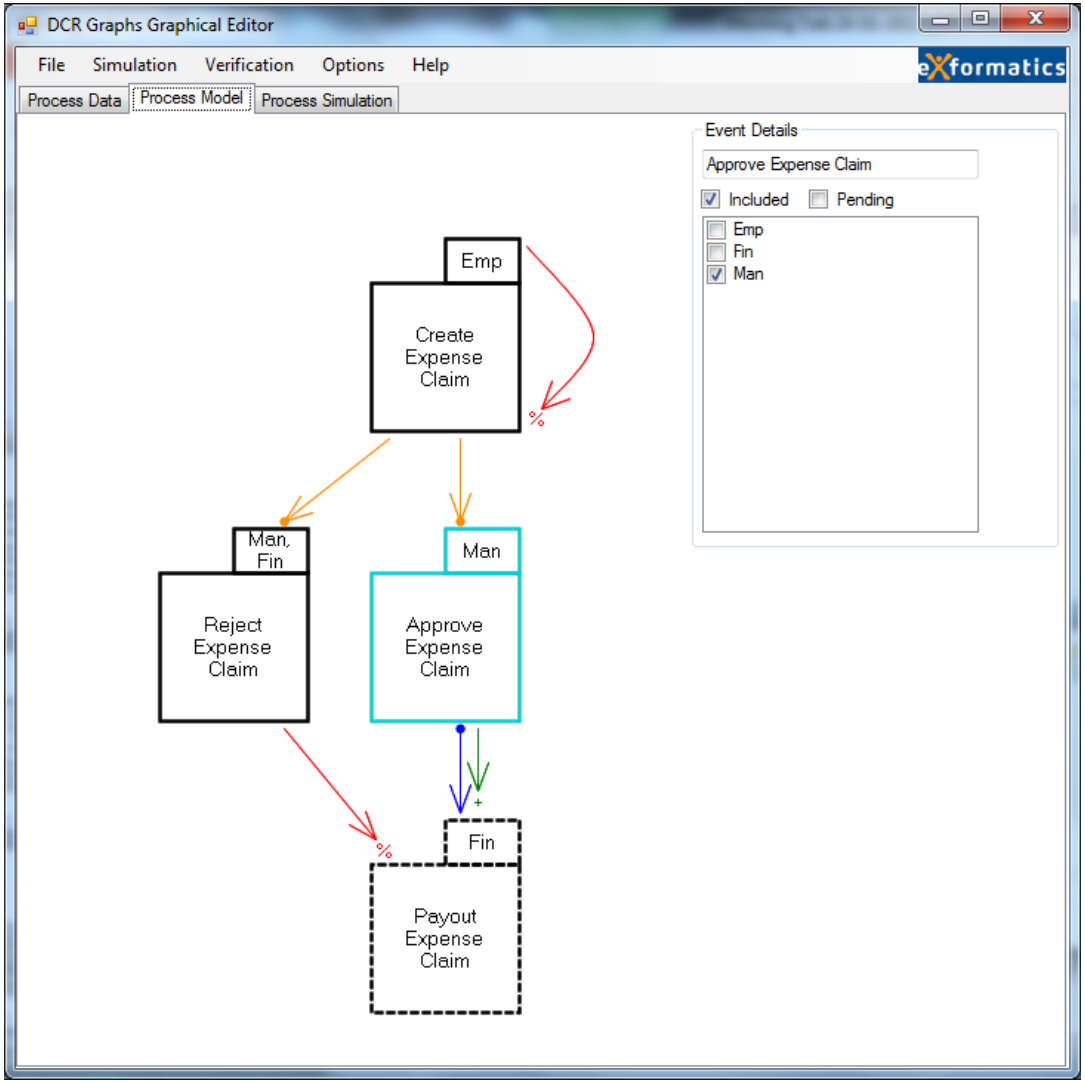
- A discrete global clock
- Execution step: an event, or a time step.
- Conditions and responses guarded by time-bounds.
  - Condition: Minimum time that should have passed since the last time an event happened.
  - Response: Deadline for when something should happen.
- Example: After approval, an expense claim should be paid out within 10 time steps.



DCR Graphs: Tools



DCR Graphs: Tools



Future Work

- Extend distributed semantics (asynchronous execution).
- Types for DCR Graphs, based on session types
- Disjunctive DCR Graphs (require one of two options as a response)
- Extend tools to handle some of the concepts discussed before

Future Work

- Explore mappings to other formalisms:
  - Declare: A declarative workflow language [Aalst et al.]
  - Guard-Stage-Milestone model: data-centric workflow model with declarative elements [Hull et al.]
  - Petri nets
- Overall goal: makes DCR Graphs more interesting commercially as it allows for interoperability with other formalisms and tools. (In particular BPMN.)

Conclusions

DCR Graphs:

- Few simple primitives, yet expressive
- Combines declarative specification with effective execution
- Support for nesting and distribution.
- Support for data and time.
- Broad application domain (healthcare workflows, case management, complex event processing)
- Positive feedback from industry
- Thank you for your time! Questions?