Towards a Unified Model for Workflow Processes

Peter Y. H. Wong
Computing Laboratory, University of Oxford.
peter.wong@comlab.ox.ac.uk

Aim

Our aim is to develop a unified model in CSP [1] for generic workflow specification, refinement and verification

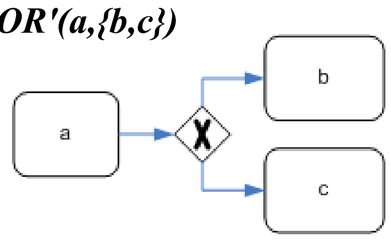
Milestones

- formalised van der Aalst's twenty control-flow patterns [2]
- applied our formalism to model some business processes defined in BPEL
- formally verified these models against abstract properties [4]
- extended our CSP models to formalise workflow choreography
- examined a real-life case-study (airline ticket reservation) in WSCI [3]

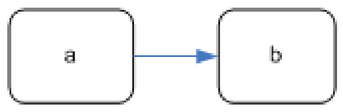
Formal Verification of Workflow Processes

Workflow Patterns

XOR'(a,{b,c})

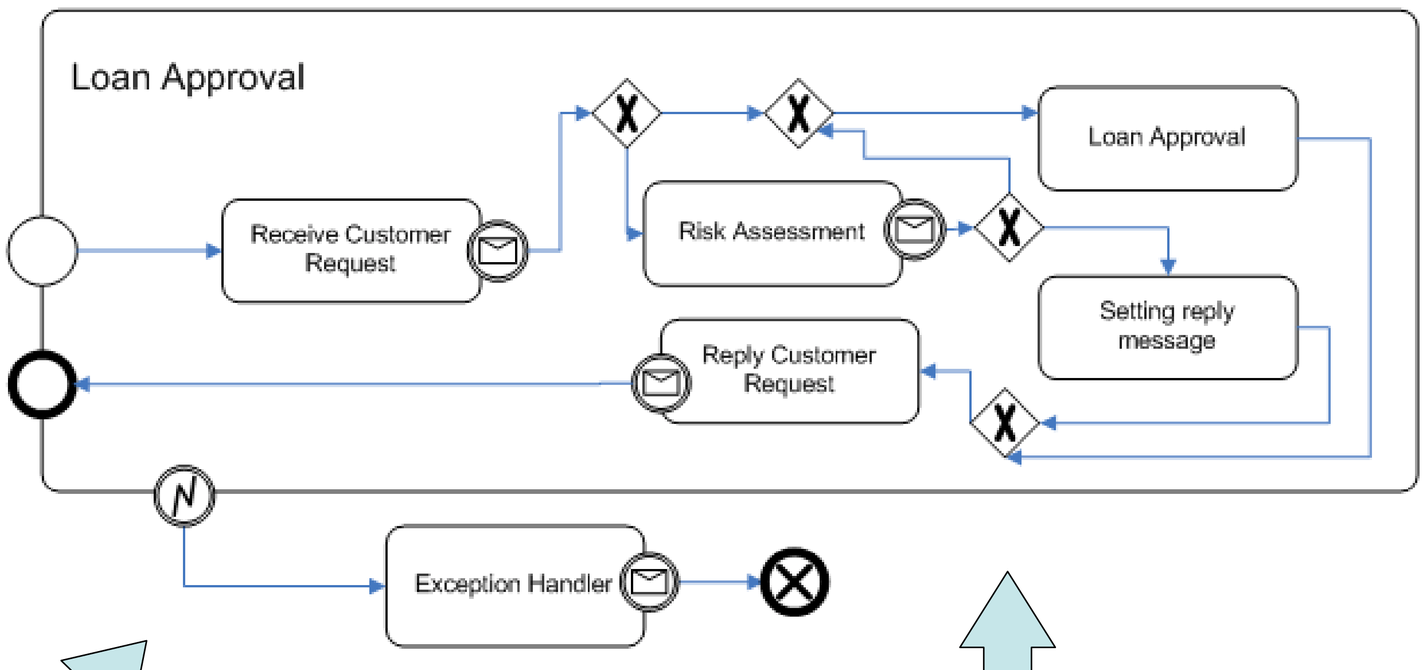


SEQ'(a,b)



SP(a,b) = *init.a* → *work.a* → *init.b* → ***SKIP***
XSP(a,X) = init.a → *work.a* → ⊓ *b* : *X* • *init.b* → ***SKIP***

SEQ'(a,b) = ***SP'(a,b)*** |[{*init.b*}]| ***SP'(b,acts)***
XOR'(a,{b,c}) = ***XSP'(a,{b,c})*** |[{*init.b,init.c*}]|
(***SP'(b,acts)*** □ ***SP'(c,acts)***)



Composed Of

models

specifies

START =
start → (*init.fault* → ***SKIP***
□ (*init.request* → (*init.fault* → ***SKIP*** □ *init.end* → ***SKIP***)))
REQUEST = *XSP'*(*request,*{*approve,assess*})
ASSESS = *XSP'*(*assess,*{*approve,accept*})
MESSAGE = *SP'*(*accept,reply*) ***FAULT*** = *SP'*(*fault,errors*)
APPROVE = *SP'*(*approve,reply*) ***REPLY*** = *SP'*(*reply,end*)
END = *init.end* → *done* → ***SKIP***
□ *init.fault* → *init.error* → *cancel* → ***SKIP***

MODEL =
let
LOAN = (***REQUEST*** |[{*init.assess*}]| ***ASSESS***)
|[{*init.accept,init.approve*}]|
((***MESSAGE*** □ ***APPROVE***) |[{*init.reply*}]| ***REPLY***)
within
(***START*** |[{*init.request,init.fault,init.end*}]| ((***LOAN*** Δ (***FAULT*** □ *done* → ***SKIP***))
|[{*init.end,init.fault,init.errors,done*}]| ***END***)) ; ***MODEL***

Verified against via FDR

$$SPEC \sqsubseteq_{FD} MODEL$$

SPEC =
let
CANCEL = *cancel* → ***SKIP***
HIGH = *init.approve* → ***REPLY***
⊓ **CANCEL**
LOW = *init.assess* → (***HIGH*** ⊓ ***CONF***)
⊓ ***CANCEL***
CONF = *init.accept* → ***REPLY***
⊓ ***CANCEL***
REPLY = *init.reply* → (***END*** ⊓ ***CANCEL***)
⊓ ***CANCEL***
END = *done* → ***SKIP***
within
start → (***HIGH*** ⊓ ***LOW***) ; ***SPEC***

Future Work

- Examine other choreography description languages
- Develop a unified model for orchestration and choreography
- Extend our model to reason about exception and compensation
- Extend our model to reason about security and mobility
- Extend our model to reason about dataflow properties
- Develop a unified model for business and scientific workflows

References

[1] C. A. R. Hoare. *Communicating Sequential Processes*. Prentice-Hall, 1985.
[2] W. M. P. van der Aalst et al. Workflow Patterns. *Distributed and Parallel Databases*, 14(3):5–51, July 2003.
[3] P. Y. H. Wong. A Case Study on the Formal Verification of Workflow Choreography, 2006.
[4] P. Y. H. Wong. A Process Algebraic Approach to Workflow Verification, 2006. Submitted for publication.