REBS – Assignment 1

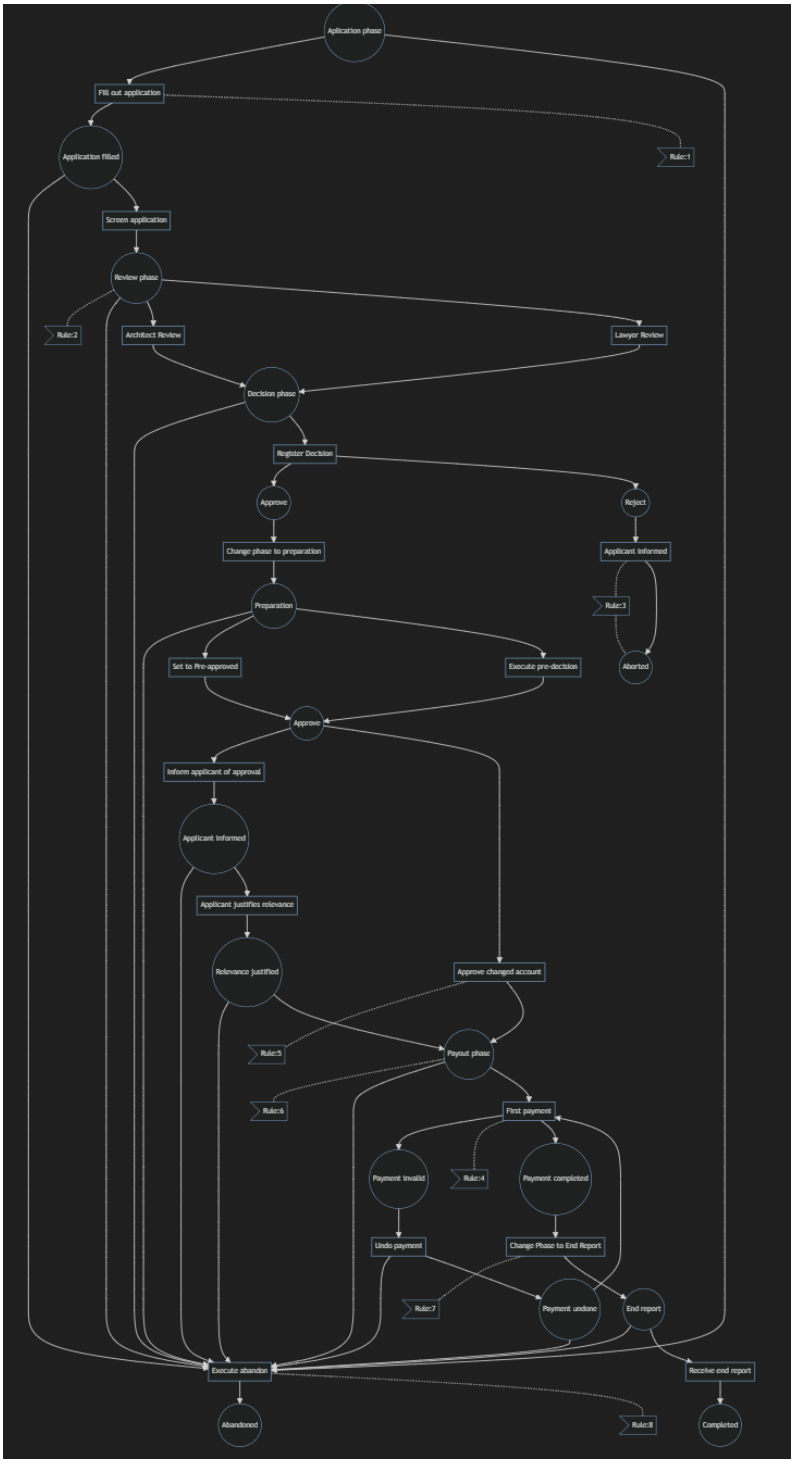
Albert Friis-Hansen (ghv657), Felix Fleckenstein (vcb456)
Group 3

8. december 2023

1 Part 1

1.1 Task 1 and 2

Note that in our model places are circles, transitions are rectangular and directed arcs are arrows. Notes to show the rules, are indeted in the leftside and showed with striped lines.



Figur 1: Petrinet

1.2 Task 3

1.2.1 Is your Petri net live and/or quasi-live?

Live means for the possibility for every transition can be reachable from any state, which is not possible in our petrinet. TO given an example, as soon as you move through the application phase, you cannot come back.

Our petrinet is quasi live, because all transitions are possible to fire through, at some point in the model, but they may not be reachable from any marking.

1.2.2 Is your Petri net bounded and/or safe?

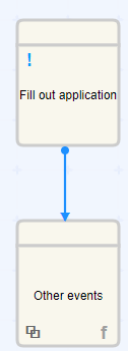
Our petrinet is bounded, because there is a limit to the number of tokens that can be in place. We can see this because our model as a logical progression with start and end points for tokens. It is not safe though because it, is not 1-bounded. In the reviewing phase, it can be possible to both need a "Architect reveiw"and a "Lawyer review". We would here end up with two tokens in "Review"transition, and decision.

1.2.3 Is your Petri net a WorkFlow net?

Our model is a workflow net, where it represents a model showing the proccess that happens when a application is submitted. Also we have a endpoint at either "Completed"or "Abandoned".

2 Part 2

2.1 Task 1

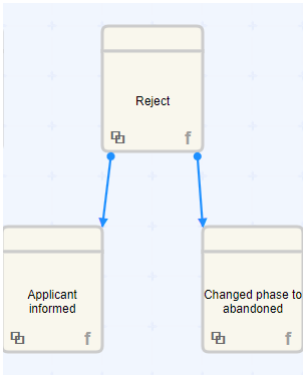


Figur 2: Graph 1

This short graph describes that the first thing that must happen, is to fill out the application. Only then may other events occur.

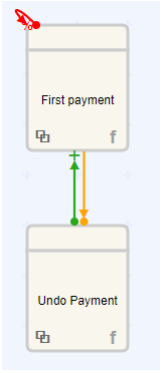
Figur 3: Graph 2

In this graph, we've modelled the relationship between the lawyer review and the architect review. These arrows means, that either excludes the other from occuring.



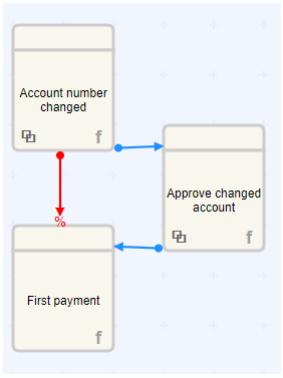
Figur 4: Graph 3

This figure describes that once we've rejected an applicant, we must eventualy both inform the applicant, and change the phase to abandoned. It doesn't matter which order it occurs in, as long as it happens after reject.



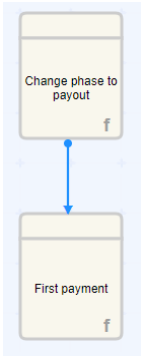
Figur 5: Graph 4

This model depicts how the payments work. We can see here that first payment exludes itself from happening once it has happened, to ensure we dont receive payment twice for the same application. The yellow (condition) arrow tells us that "Undo Payment"cannot happen before first payment has happened at least once. The green (include) arrow means that if "Undo Payment"happens, then first payment must be the next step.



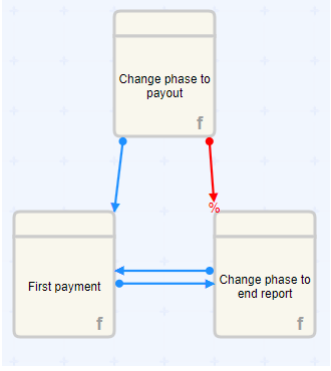
Figur 6: Graph 5

This model tells us that the first payment cannot happen before the account number is changed and approved.



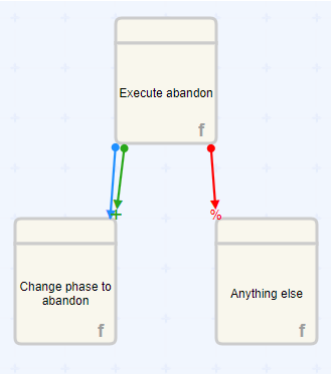
Figur 7: Graph 6

This figure tells us that if we change the phase to payout, then we must eventually receive first payment.



Figur 8: Graph 7

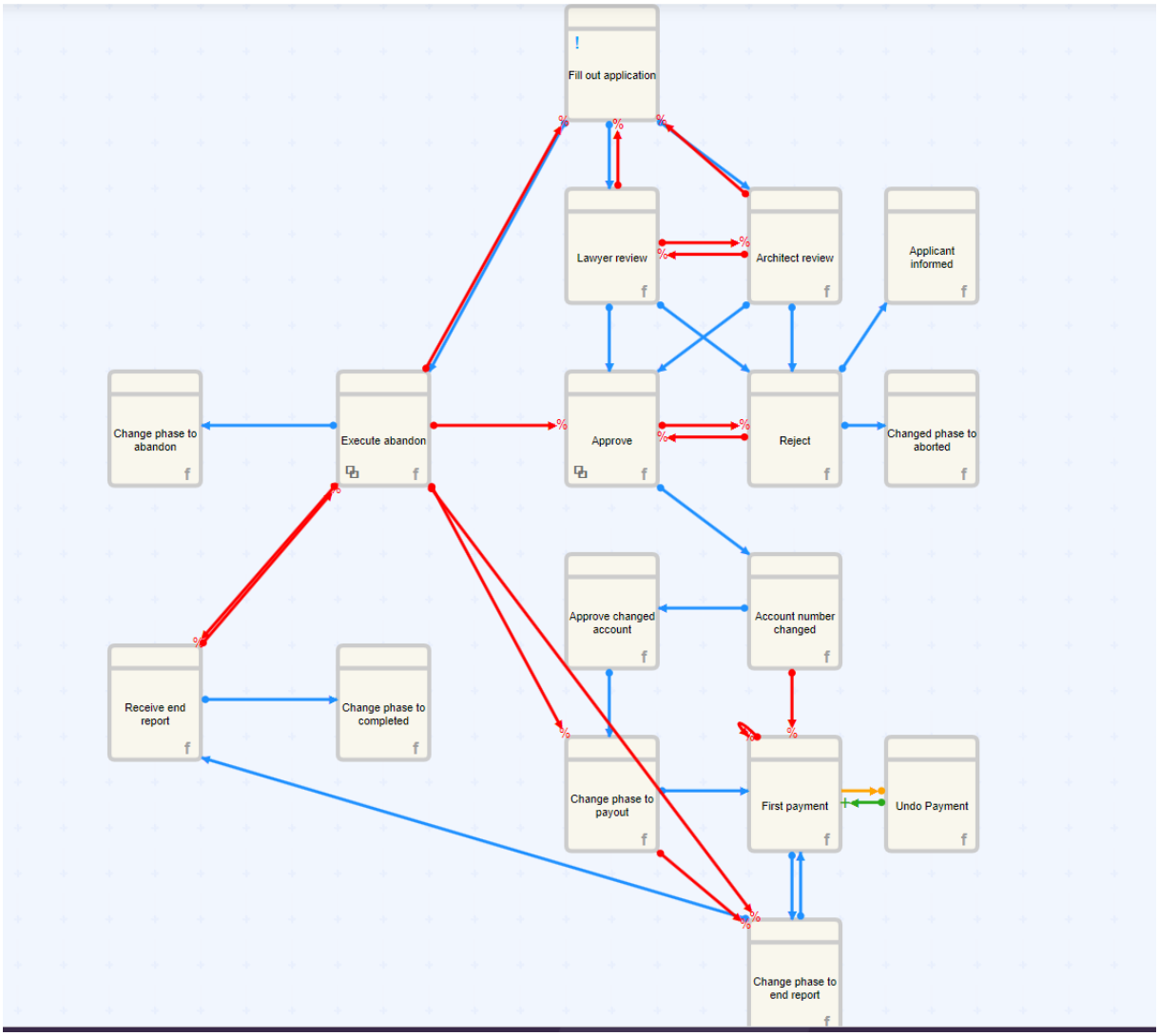
This graph describes that we cannot change the phase to end report before we've received the first payment.



Figur 9: Graph 8

This graph tells us that if we execute abandon, the only thing we can do is to change the phase to abandon.

2.2 Task 2



Figur 10: Graph 1-8 combined into a full DCR graph

2.3 Task 3

2.3.1 Which relations did you not use in your models?

- **No response**
- **Value**
- **Spawn**
- **Milestone**

2.3.2 Could some of the rules have been modelled in more than one way? If so, give one or two examples.

We could've modelled the beginning of the DCR graph differently, using milestone. If we'd have done this, we would start the milestone in "Fill out application", so that it's the only way we can start the process. We could also have "saved"some arrows by nesting some of the activities. Especially by nesting everything else than execute abandon and change phase to abandon, so we'd have less arrows pointing to execute abandon.

2.3.3 How does your model differ from the Petri net? Do they exhibit the same language?

They don't exhibit the same language, as the DCR graph uses a graphical notation with nodes representing events, conditions, responses and milestones. The petri net is a more mathematical notation, which consists of places, transitions, arcs and tokens. They are both used to model dynamic systems, but they employ different languages and notation.