Case studies

In this document we present the case studies benchmark that we have considered to perform the evaluation of our proposal. These case studies have been taken from the literature, in particular from [EsVa16]. More specifically, in https://github.com/pvgorp/AD2SC/tree/master/grgen2sc/input, the authors provide the case studies in .xmi or in .uml2 extension format. Taking these activity diagrams as starting point, we have carry out several common changes in order to be able to apply our proposal. Next we present the common changes we have made to the considered case studies.

Firstly, although in the graphical representation it seems that the model has fork and join nodes, they have been internally defined as fork node elements (there is no join node elements). Thus, we have had to transform fork to join nodes where corresponds.

Secondly, we have also changed the model so that it does **not contain loops and nested synchronization elements** (so that each fork has only one associated join).

Finally, we have applied our stereotype <<monitored>> to specific object nodes depending on the concrete test.

In the **Appendix** we provide, for all the case studies, the complete source models together with the resulted models after having made the changes. Since the tests presented in the paper are performed twice; (1) one monitoring the half of the object nodes and (2) another one monitoring all the object nodes, in the changed models we have also filled with a different color the object notes monitored in cases (1). We use the numeration of the paper, that is, since process P1 corresponds to the Obesity protocol, we start with P2.

Next we present these case studies and some of their particularities.

[**EsVa16**] R. Eshuis and P. Van Gorp, “Synthesizing object life cycles from business process models,” Software & systems modeling, vol. 15, iss. 1, pp. 281-302, 2016.

P2. Bikeshop

This case study corresponds to a real-life industrial process modeled by [EsVa16]. More specifically, it represents the ordering and delivery of bikes.

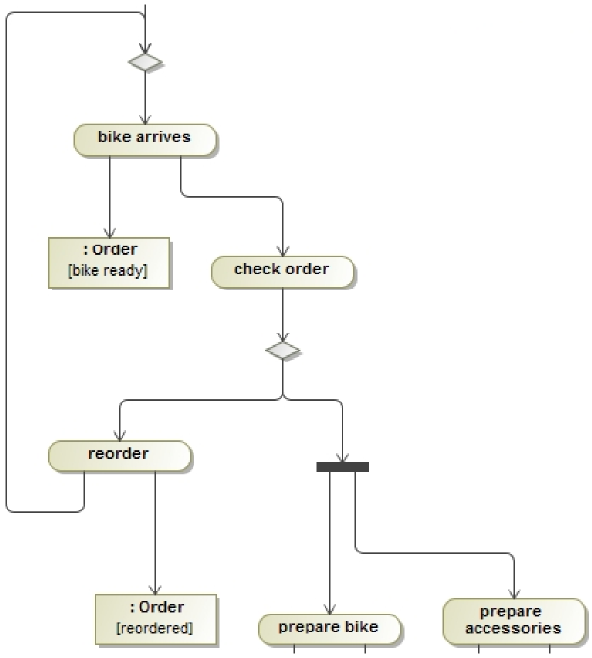
1.1 Source model

The source model can be seen in Appendix. It is characterized by being block-structured, that is, each fork that starts parallel branches has one matching join that synchronizes the branches.

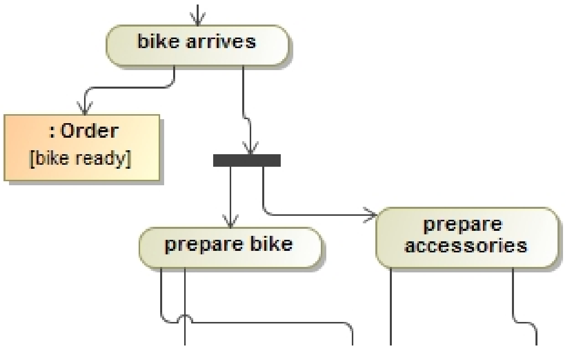
1.2 Changes made to the source model

This model contains a cycle from action node “reorder” to merge node targeting in activity node “bike arrives”. We have modified the model so that no cycle is presented, that is, we consider the fact that, when bike arrives, no order check is performed, but bike and bike’s accessories are prepared.

Source model extract



Changed model extract



P3. Dermatology

This case study, as well as the previous case, corresponds to another real-life industrial process which particularly models the handling of dermatology patients.

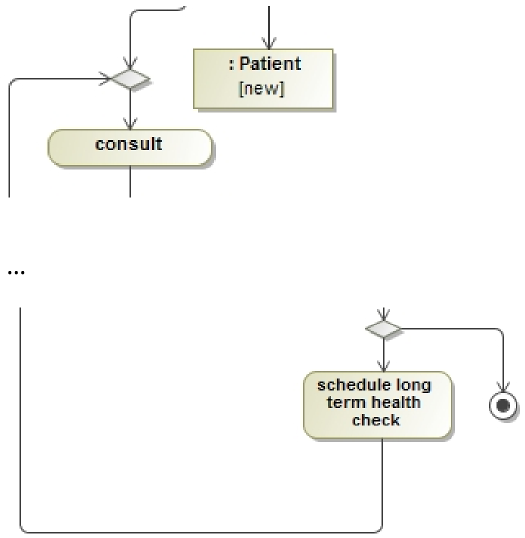
1.1 Source model

As in the previous case, this case study is also characterized by being block-structured (the source model can be seen in Appendix). In particular, it contains two fork-join parallel branches.

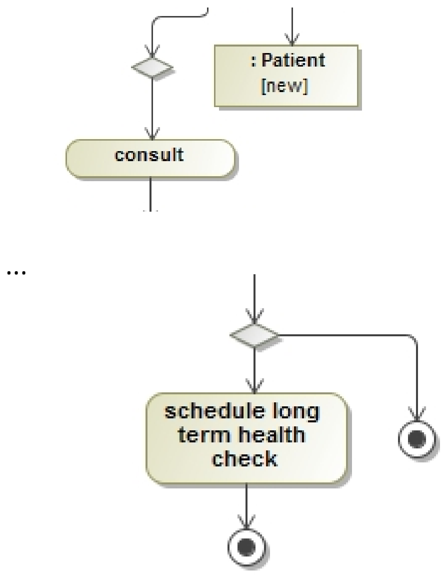
1.2 Changes made to the source model

As in the previous case, this model contains a cycle from action node “schedule long term health check” to merge node targeting in activity node “consult”. We have modified the model so that no cycle is presented.

Source model extract



Changed model extract



P4. Create catalog

This case study contains an activity diagram with stateful object nodes that [EsVa16] takes from literature.

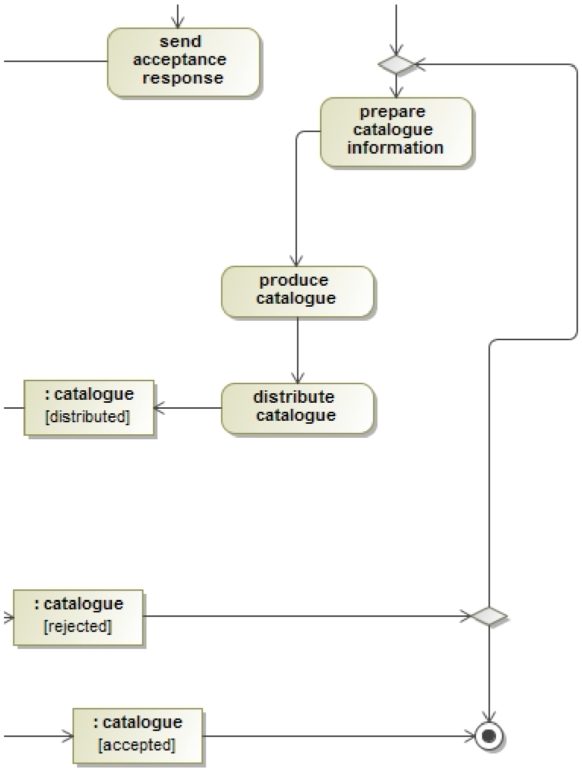
1.1 Source model

As we can see in Appendix, the model is characterized by a simple fork-join parallel branch.

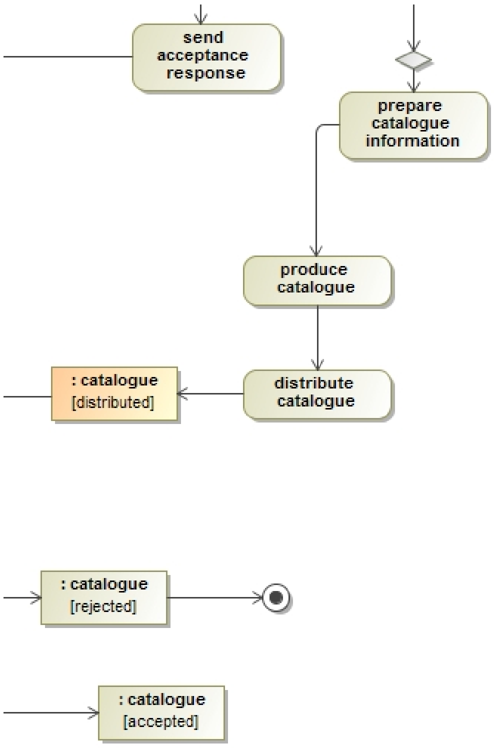
1.2 Changes made to the source model

As in the previous case, this model contains a cycle, which we have modified.

Source model extract



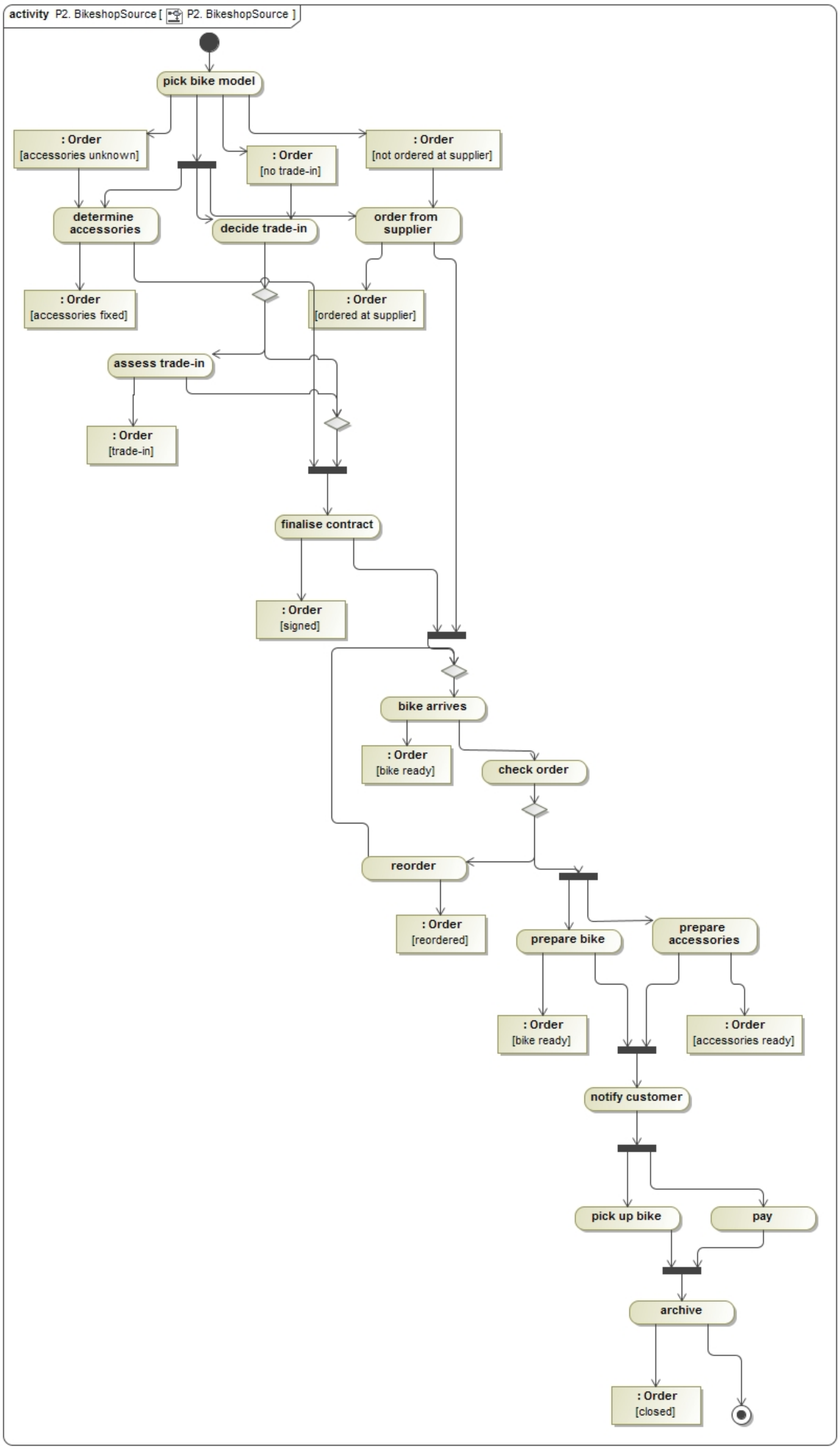
Changed model extract



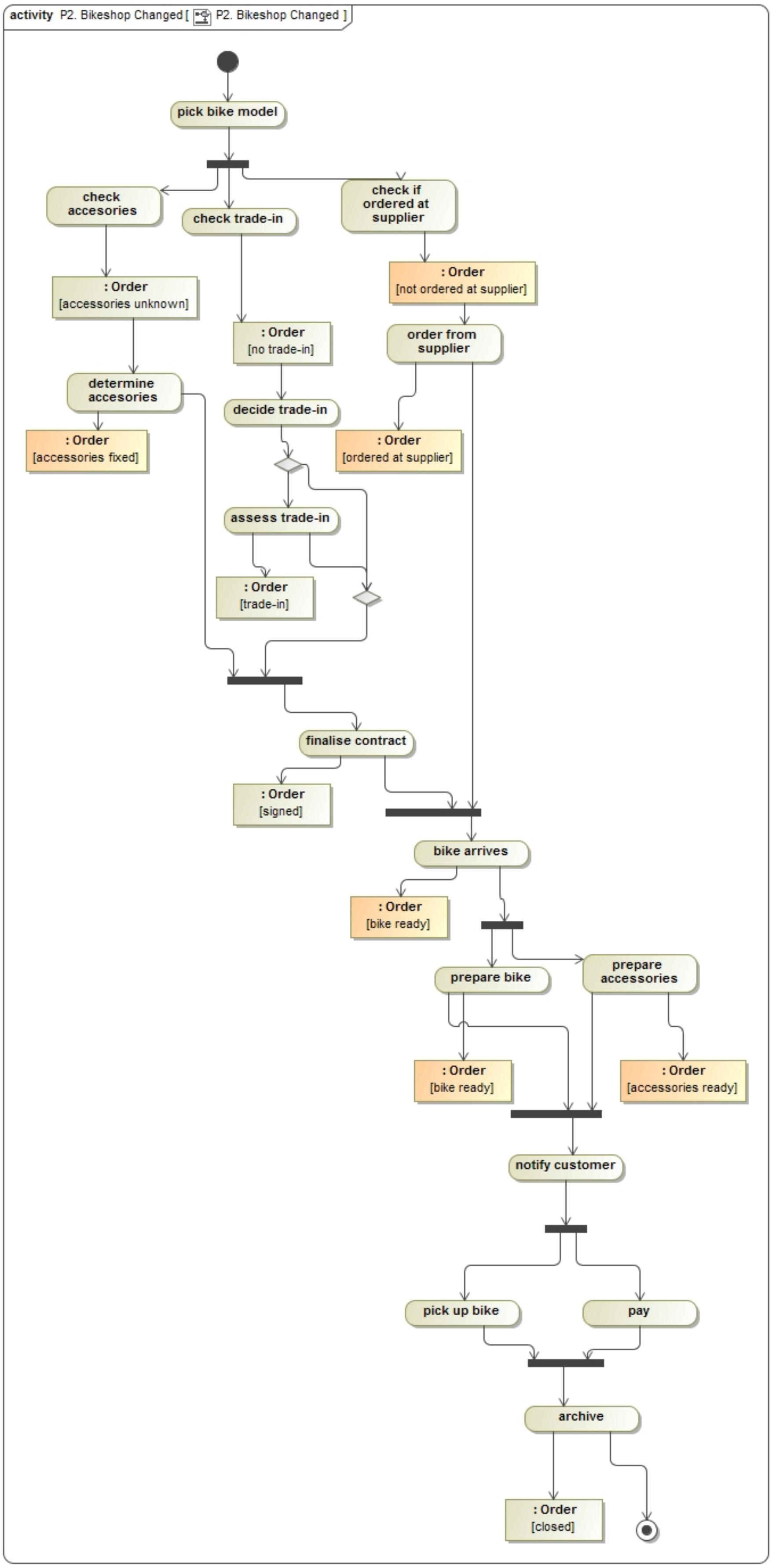
Appendix

P2. Bikeshop

Source model

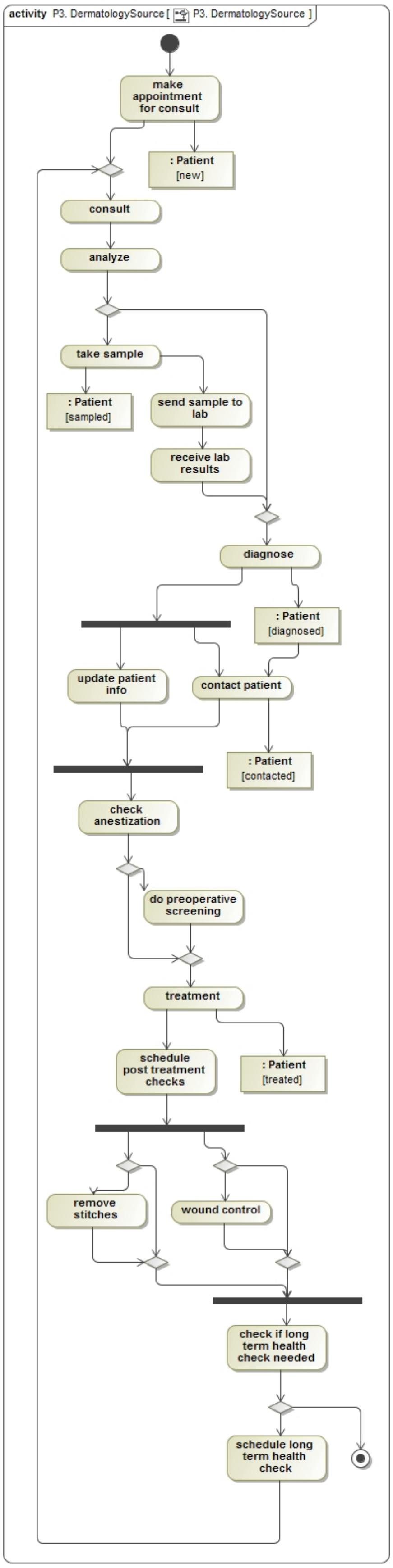


Changed model

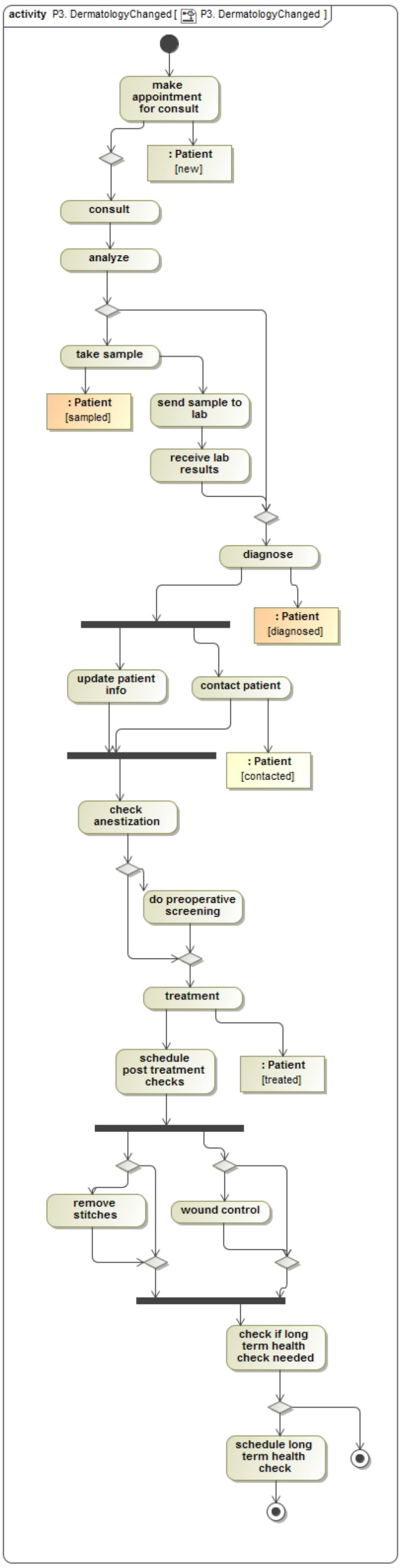


P3. Dermatology

Source model

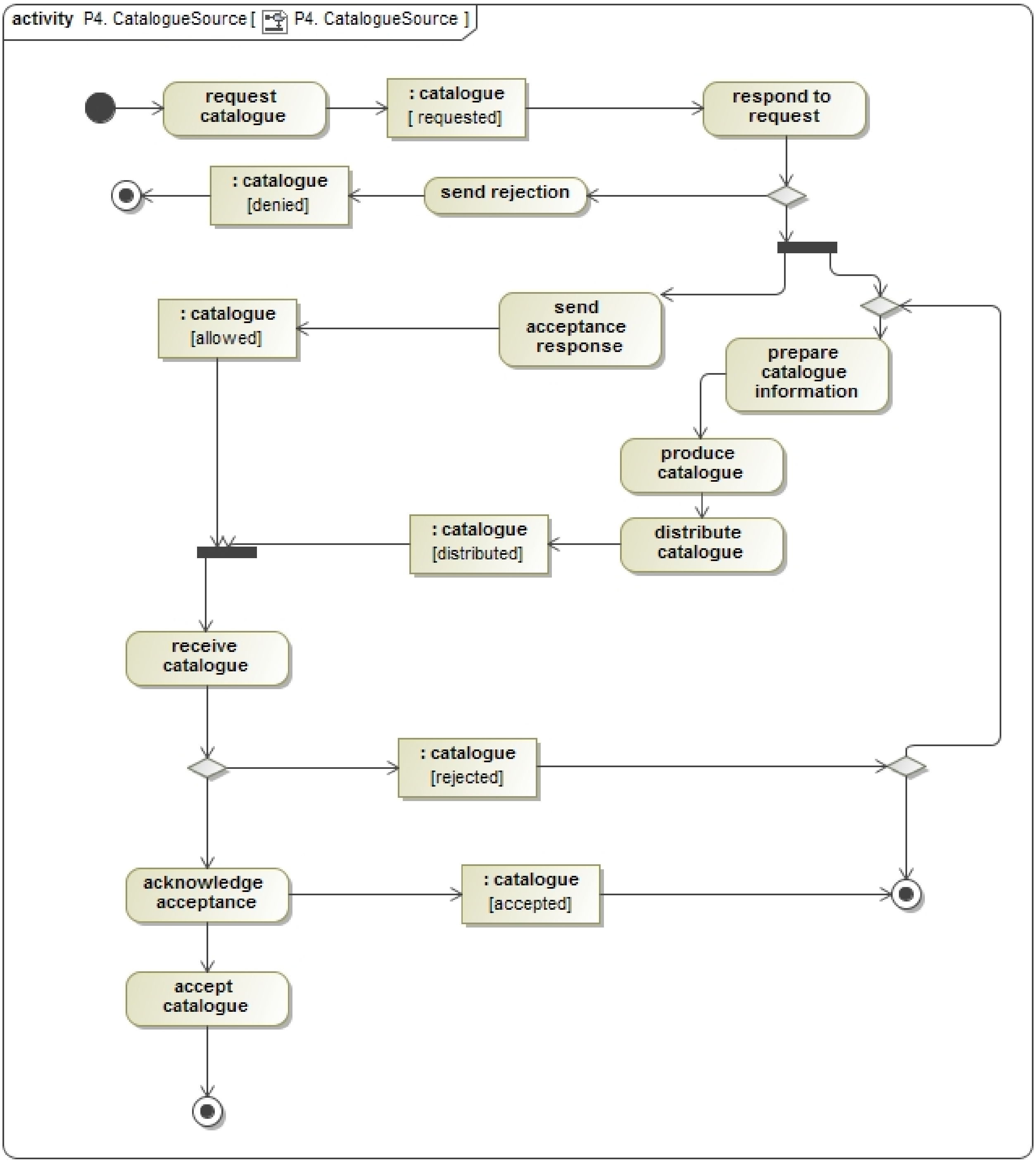


Changed model

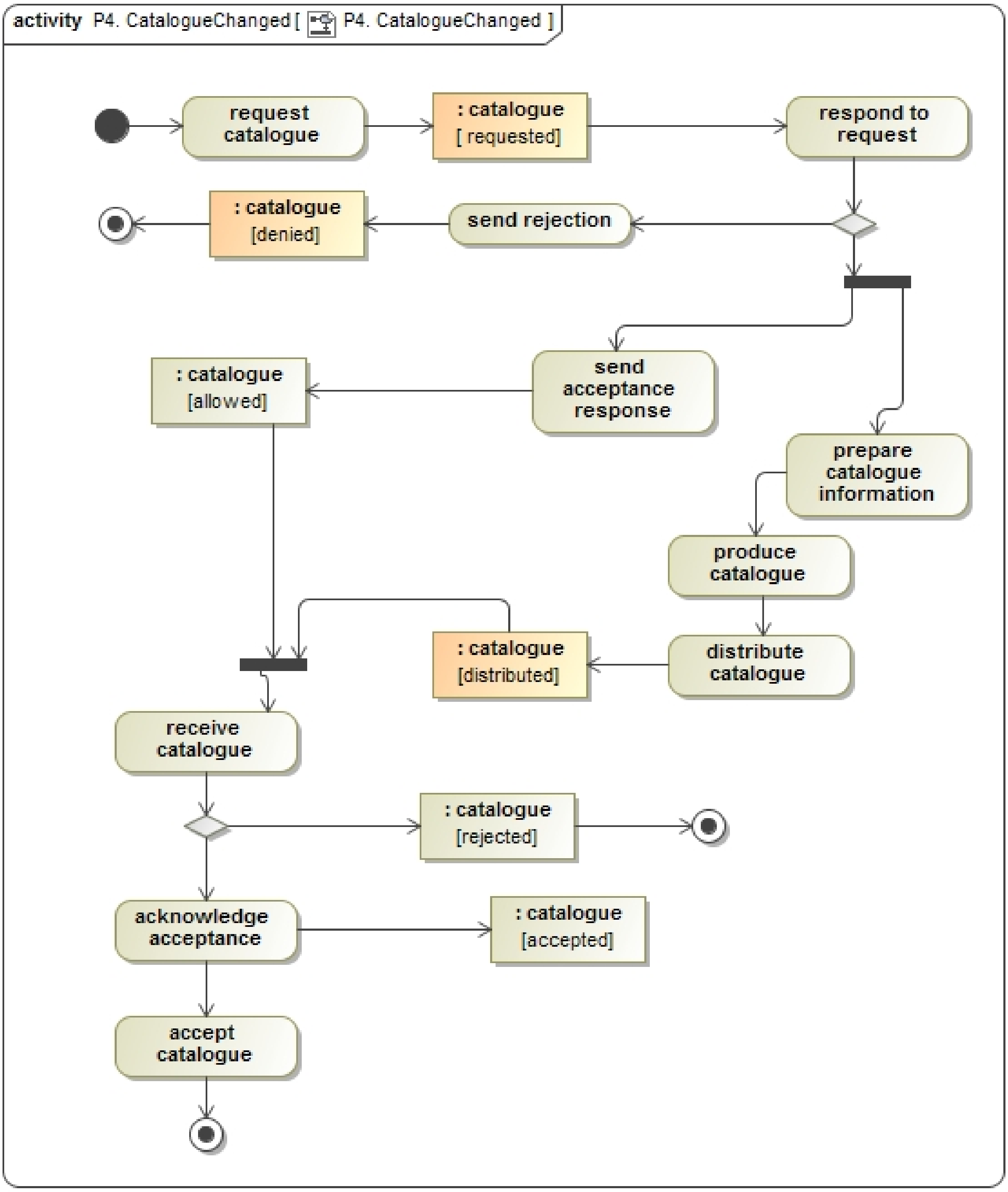


P4. Create catalogue

Source model



Changed model


activity P4. CatalogueChanged [P4. CatalogueChanged]
request catalogue
: catalogue [requested]
respond to request
send rejection
: catalogue [denied]
send acceptance response
: catalogue [allowed]
prepare catalogue information
produce catalogue
distribute catalogue
: catalogue [distributed]
receive catalogue
: catalogue [rejected]
acknowledge acceptance
: catalogue [accepted]
accept catalogue