ProVer (Process model Verifier)User Guide

Hakan METIN

February 27, 2016

1 Overview

Investigations on some process repositories demonstrated that between 10% [Mendling 09] to 50% [Vanhatalo 07] of process models are flawed and contain inconsistencies. In some critical businesses, having unsound process models could be very harmful for the quality of the delivered services and products. Inconsistencies can range from very basic syntactical errors, to more complex issues such as deadlocks, inconsistent allocation of resources and time duration on process's activities or any proprietary business constraints that might be violated by potential process executions (paths). Some process models, depending on the business domain (healthcare, military, etc.), can be very complex and may contain more than 250 activities with very sophisticated control and data flows, resource and time constraints [Christov 14][Simidchieva 10]. Trying to analyze these processes and to verify them without the help of a tool would be unmanageable. In [Mendling 09], a study demonstrated that over 2000 inspected process models, 10% of these models where unsound. An equivalent study on SAP repository containing more than 600 complex process models demonstrated that more than 20% had flaws. [Mendling 06, 07]. Similarly, Gruhn et al. [Gruhn 07] collected 285 EPC (Even-Driven Process Chains) from different sources i.e., repositories, scientific papers, thesis, etc., and came to the conclusion that even though process models were syntactically correct, more than 38% of them were unsound. Finally, Vanhatalo et al. [Vanhatalo 07] analyzed more than 340 Business process models with a focus and the control flow aspect to realize that half of them were invalid i.e., contained deadlocks or unreachable activities. To avoid this kind of errors, formal verification on these models is needed. Properties implanted in ProVer allows anyone to verify his model very easily by simply choose properties on ProVer graphical mode.

ProVer can be used to verify three classes of properties: soundness properties, resource constraint properties and business rule properties. The soundness properties include termination along all paths, termination along at least one path, termination along a given path and reachability of a, ActivityFinal node. The resource constraint properties include constraints on execution time, manpower for different roles required for different activity classes. The business rule constraints include arbitrary constraints specific

to the process model domain such as a limit on the number of time an activity of a given class must be executed.

ProVer is an eclipse plugin which use an SMT solver witch is z3 (Microsoft Research MIT License) This tool uses UML Designer by OBEO to visualize process diagrams. User simply chooses the properties to be verified and launch the verification.

ProVer can be integrated to the MERGE multi-concern system engineering platform. It is open source and distributes under EPL (Eclipse Public Licence)

2 Properties implanted on ProVer

A categorization of the different properties that can be expressed on software process models is presented on the following table. It represents the outcome of a literature review in the business process domain and in software methods and practices.

Category	Definition
(1) Soundness	
OptionToComplete	A started process can always complete
ProperCompletion	No other activity should be running when the process terminates
NoDeadTransition	All the activities must be reachable
Soundness with data	
MissingData	The data are always present when they need to be accessed *(e.g. no data missing to start an activity)*
UselessData	The data created are always used *(e.g. no data created but never used before the process ends)*
InconsistentData	The data can never be in an inconsistent state *(e.g. no data modified by multiple activities in parallel)*
(2) Organizational	
InTime	There is enough time to perform the activities *(e.g. the process will terminate before X hours/days)*
MissingResource	No missing resource to start an activity *(e.g. there are enough agents to do the process)*
(3) Business	
ExistenceActivity	`A` is executed more/less/(between) X (and Y) times
ExistenceTimeActivity	`A` is executed before/after/(between) `X` (and `Y`) time unit
ExistenceTimeData	`ArtefactA` is available before/after/(between) X (and `Y`) time unit
ExistenceTimeResource	`ResourceA` is used before/after/(between) X (and `Y`) time unit
Relation	`A` is executed before/after/in-parallel/in-exclusion/(between) `B` (and `C`)
RelationData	`ArtefactA` is available before/after/in-exclusion of `ArtefactB`
RelationActivityData	`ArtefactA` is available before/after/in-parallel/in-exclusion/(between) the execution of `B` (and `C`)
...	...

Table 1: Overview of the software properties we identified

`Option to complete` is a mandatory property, it has no sense to not finish a process model.

3 Plug-in Installation

A prerequisite for installing ProVer is the installation of MERgE platform 3.0.1. Two mechanisms are offered to install the plug-in on MERgE platform.

3.1 Installation - Update Site

1. Open Merge platform and go to `Help` → `Install New Software...`

2. This shall open a new window as shown in figure . Click on the `Add...` Button. This shall open an Add Repository dialog box. Add the following informations and Click on `OK`.

Name *ProVer*

Location *http://merge.lip6.fr/MERGE_PROVER*

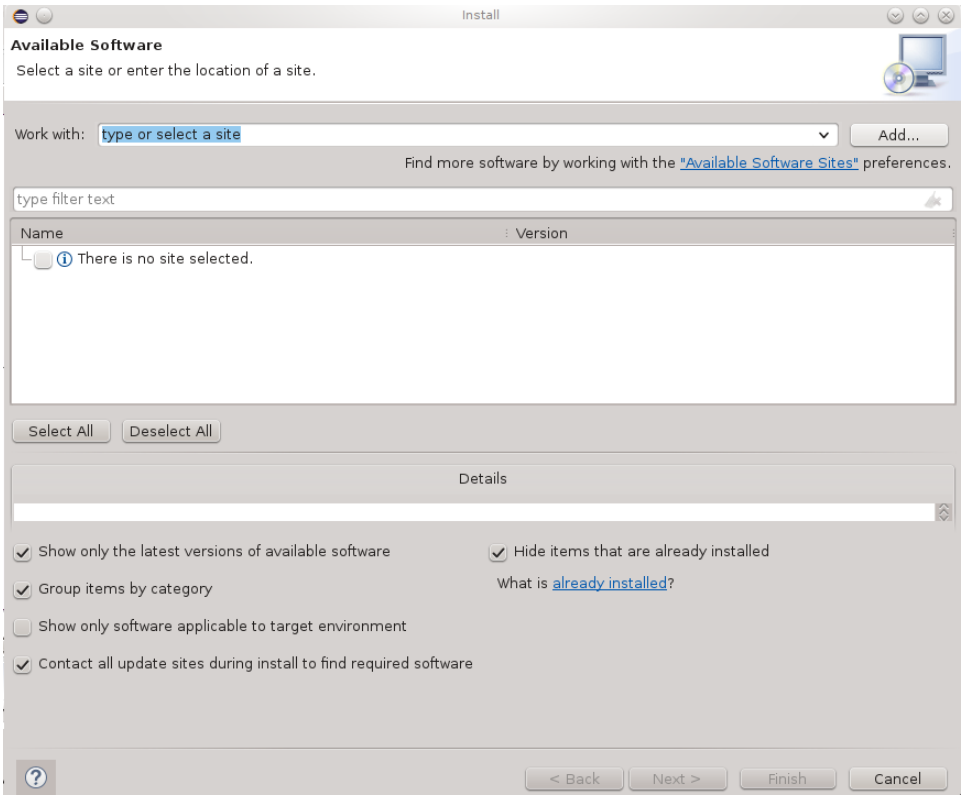

Figure 1: Installing the Plug-in - update site

3. Select the plug-in `Prover` and Click on `Next`. Click on `Next` once again for the installation details.

4. Accept the license agreement and Click on `Finish`.

3.2 Installation - Local Archive

1. The plug-in is archived in a file named ProVer.zip

2. Open MERgE platform, and go `Help` → `Install New Software`, this shall open up a window for plug-in installation

3. Click on `Add`, this shall open an `Add Repository` dialog box.

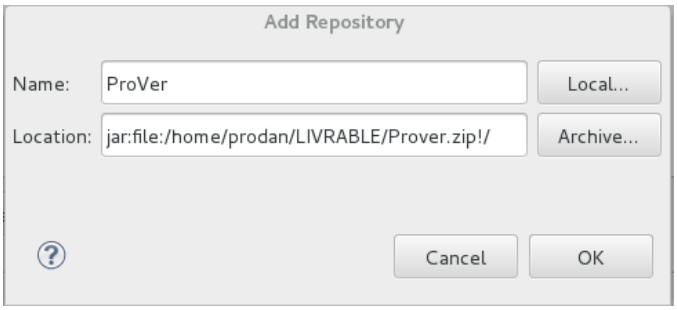

Figure 2: Installing the plug-in - archive file selection

4. Click on Archive and browse ProVer archive File (see figure 2) and click on `OK`

5. Uncheck the option `Group by category`

6. Select the plug-in and click on `Next`, click in `Next` once again for the installation details.

7. Accept the license agreement and Click on `Finish`.

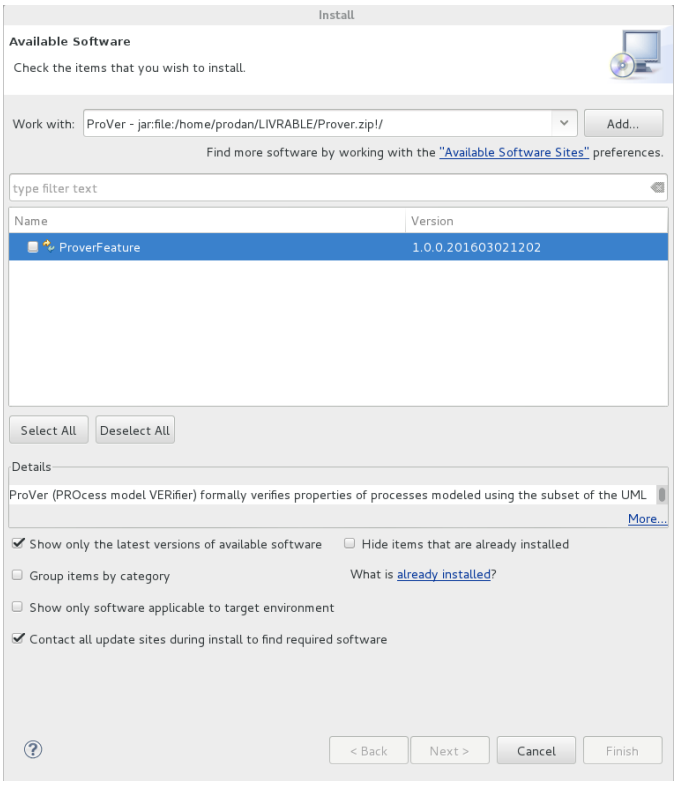

Figure 3: Installing the plug-in - local archive

4 Tool Layout

ProVer is deployed as an eclipse plug-in, It is composed of different views. Each of these is detailed in the following

4.1 Process Editor

The used process editor is `UML Designer` provide by Obeo. It allows the development of process models using UML 2.4 activity diagram.The editor allows adding activities, actions, initial node, final node, and other control flow nodes like decision, merge, fork and join nodes.

ProVer uses UML Opaque Actions to model atomic activities of a process model that can not be decomposed into sub activities. The description of an opaque action describes its inner implementation. It can be added to the action using its "body" property under the semantics tab of properties view. Different roles associated with the process model can be modeled using partitions in activity diagram, which is a similar concept as swimlanes in BPMN. We now present different view of this tool.

4.2 Main Verification view - Process tab

ProVer main view allows to launch verification of selected process model.

In every process we want to verify the completion i.e reach an ActivityFinal node in different mode i) `Universal`: all existing path reach an ActivityFinal node. ii) `Existential` It exist at least one path witch reach an ActivityFinal node.

Another important configuration is the BMC Limit. Effectively the formal method used to verify the process model is Bounded Model Checking (BMC) with Solver Modulo Theory (SMT) solver. This technique checks if all properties are satisfied until this bound. The only SMT solver available on this tool is z3 (MIT License, Microsoft Research) And the last configuration possible is dead transition that means a node is never reachable. We can see an example of configuration in figure 4.

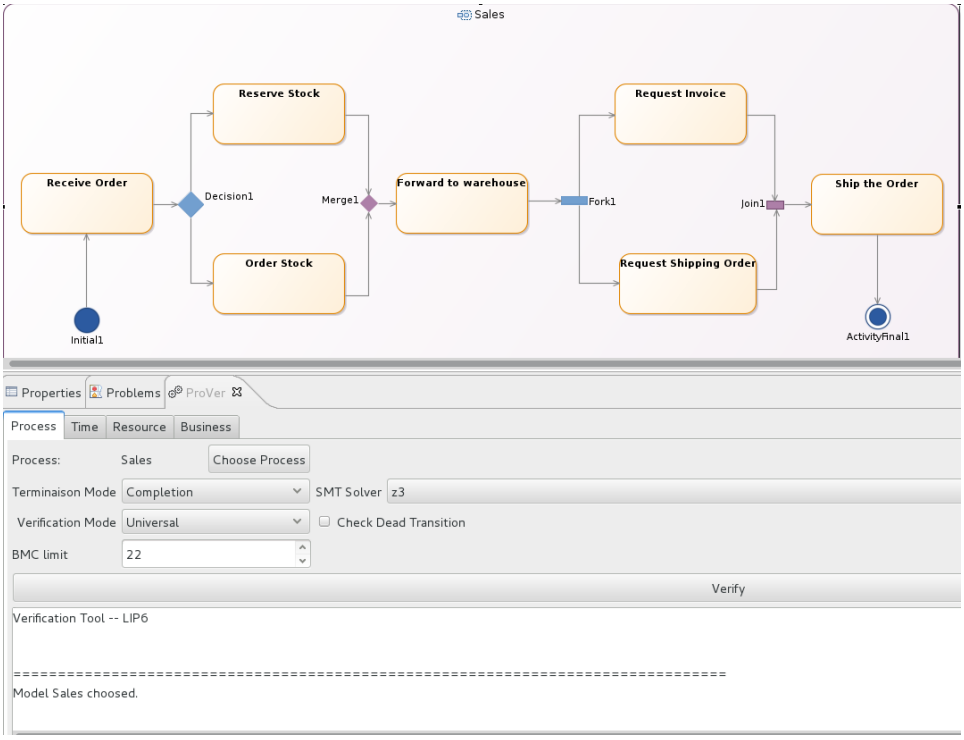

Figure 4: Tool overview

4.3 Time tab

Often, each activity is assigned with a duration and we have a fixed duration to finish the entire process model. ProVer allows checking this property automatically. We can affect a duration to each action and a global time.

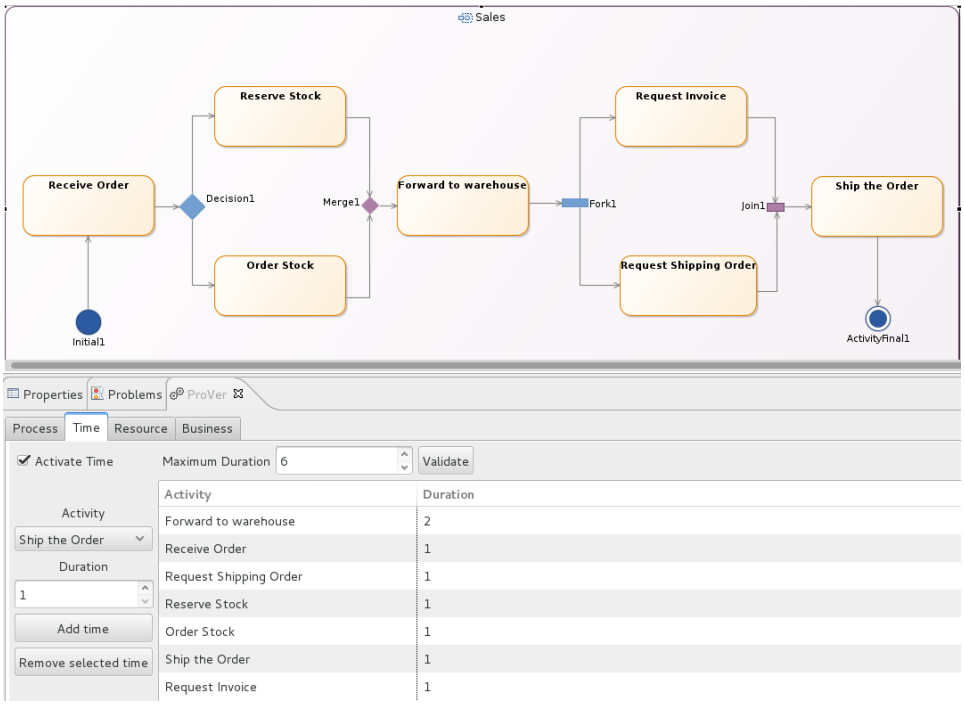

Figure 5: Time tab

As one can see in figure 5, we affect each activity one time unit expect `Forward to warehouse` activity wich assigned with two time units. The global time assigned to this process is 6 time units. Prover verifies the process can be reach an ActivityFinal node in "global time" with affected time units to each activities (`In Time` property)

Here detailed presentation of this view

- `Activate Time`: Enable or disable time constraint verification
- `Validate`: Validate maximum time to use during process
- `Add Time`:Add a new time constraint to a selected activity
- `Remove selected time`: Delete a time constraint for a selected activity

4.4 Resource Tab

This tab allows specifying resource constraints associated to process model activities. Such resource constraint are specified by a constraint name defining the resource class and an integer defining the maximum cardinality for instances of this class allocated to the constrained activity.

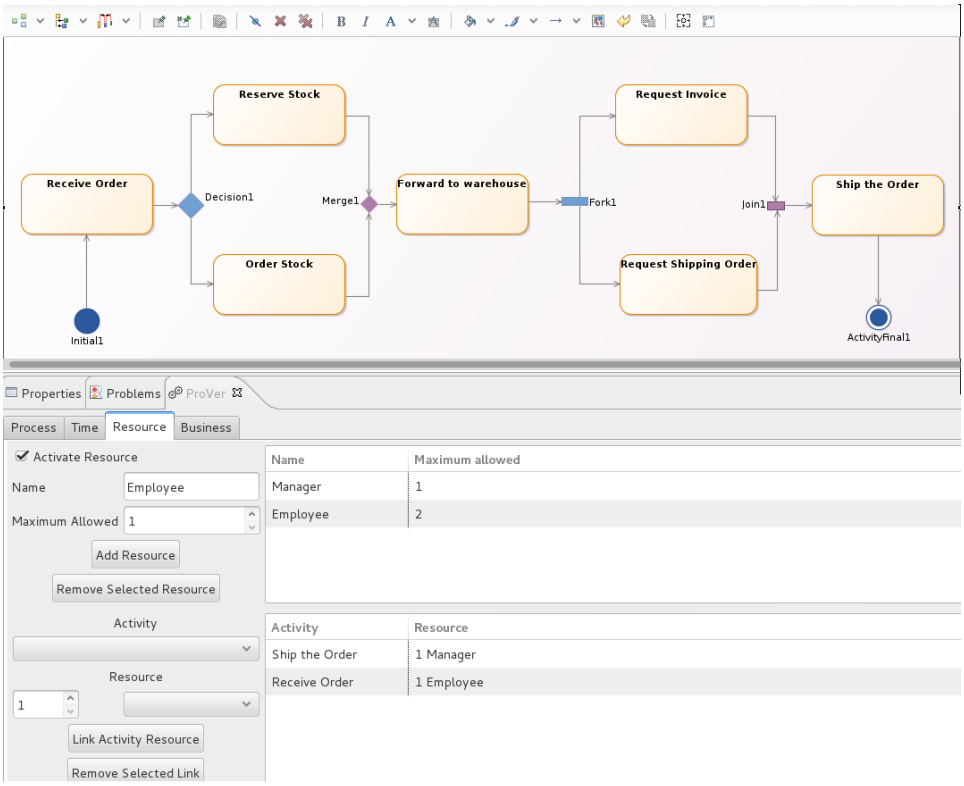

Figure 6: Resource tab

In figure 6, we can see a configuration of resources (Manager and Employee) where there is at most 1 manager and 2 employees `Ship the order` was affected to manager and `Receive Order` was affected to 1 employee. Many resources can be assigned to each activities.

Here detailed presentation of this view

- `Activate Resource`: Enable or disable verification for the resource constraints specified
- `Add Resource`: Create a number of resource witch allow using on parallel
- `Remove selected resource`: Remove selected resource
- `Link Activity Resource`: Create a resource constraint by linking a created resource to a selected activity.
- `Remove selected link`:Remove a resource constraint by unlinking a created resource with an activity.

4.5 Business Tab

This tab allows specifying business constraints to be verified by the process model. Different kind of business properties exists:

- Relational: Before, After, in Parallel, in Exclusion
- Occurrence: Selected activity must be executed at least N times and/or at most M times before the process enactment terminates.

- Time usage: Selected Activity or Resource must be used before / after a time T

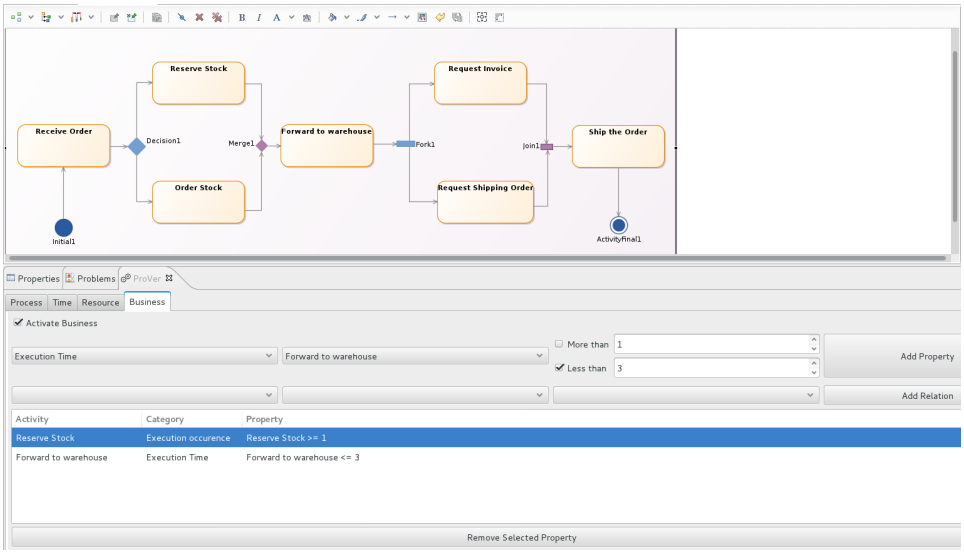

Figure 7: Business tab

We can see an example in figure 7, two business properties are added

- `Reserve Stock` must be executed at least one times, `Forward to warehouse` must be execute before global time inferior or equal to 3

Here detailed presentation of this view

- Activity execution occurrences:
- Activity execution time: selected activity A is executed at least Tmin and/or at most Tmax time units
- Resource usage time: selected resource is used at least Tmin and/or at most Tmax time units
- Data availability time: data on selected activity output pin P is available at least Tmin and/or at most Tmax time units.
- Relation Before: Selected activity A is executed before selected activity B
- Relation After: Selected activity A is executed after selected activity B
- Relation Parallel: Selected activities A and B must be executed in parallel
- Relation Exclusion: Selected activities A and B are not both executed during enactment. Either A is executed by B is not, or B is executed and A is not, or neither A nor B are executed.

5 Create a model

We now explain how to create a model.

1. Choose `File` → `New` → `Other...` This shall open a wizard
2. In this window, choose `UML Project` in `UML Designer` folder and click on `Next` (see figure 8)

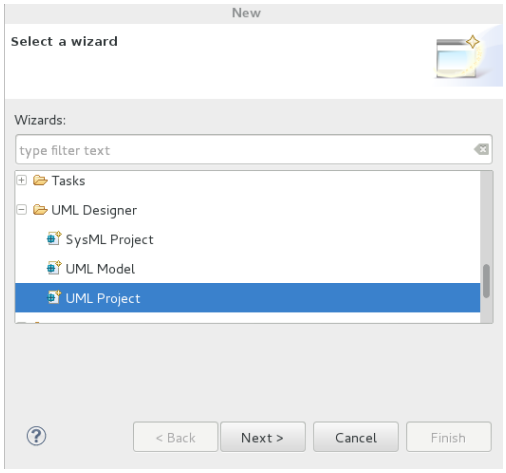

Figure 8: Creation UML project

3. Enter your project name and click on `Finish`
4. Eclipse proposes to open Modelling perspective, click on Yes. If you click on No you can open the perspective by clicking on at the top right of the `Open Perspective` Button, this shall open a window and choose `Modelling`. (Eclipse must be installed with modeling package in eclipse website).
5. On `Dashboard` click on on Activity Diagram
6. This shall open diagram window and you can then edit the activity diagram which properties you want to subsequently specify and verify using ProVer

5.1 Import Model

We can also import an exported model.

1. Go to `File` → `Import` This shall open a window
2. In this dialog box, choose `Existing Projects into Workspace` in `General` (see figure 9)
3. Browse the folder hierarchy to find the file to import, select it and then click on finish.

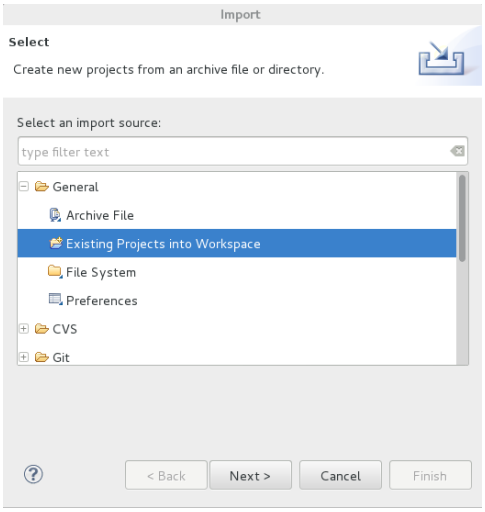

Figure 9: Import new project

6 Example

In this section we present step by step examples of process model verification using ProVer. In this first example we specify and then verify some properties which are `Completion` (mandatory) and `In Time` (defined by selected time units) on the Sales process model shown in figure 4 and available to be imported from:(http://merge.lip6.fr/SalesExampleUML.zip) This activity is composed by 7 actions which describe a sales.

1. Open the diagram representation
2. Open the verification UI window by choosing `Window` → `Show View` → `Other`
3. Choose the ProVer option from Prover menu.
4. Click on on process, this shall write the name of the process on ProVer. (see figure 10)
5. We can click on chosen process.
6. User now specify the process model properties which are `Completion` (mandatory) and `In Time` (defined by selected time units see figure 11).
7. we can see in figure 11 all affected times and selected maximum duration of the process model.

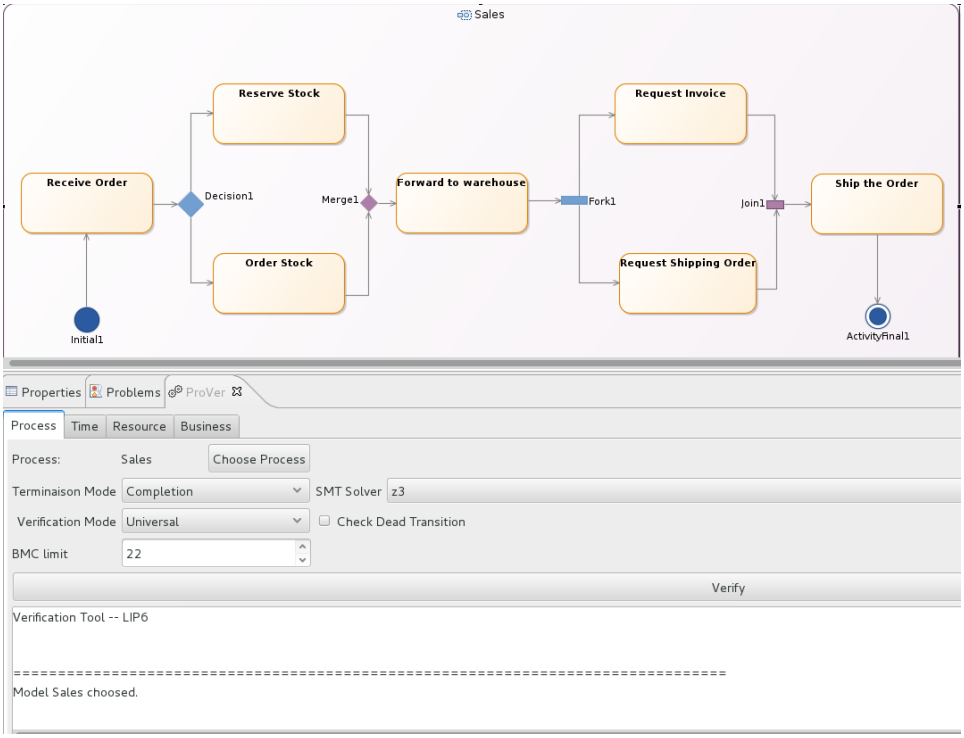

Figure 10: Choose Diagram

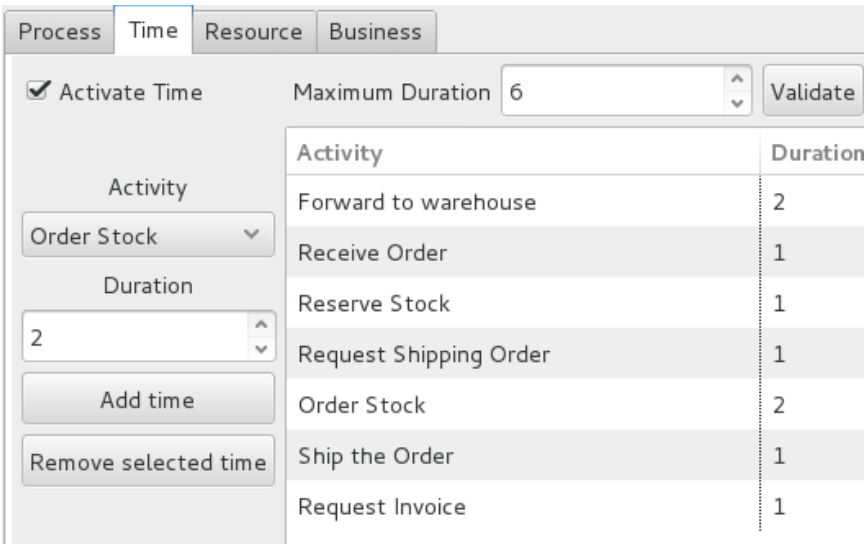

Figure 11: Time Properties

6.1 Result

This section present result of defined properties above. Verification with `universal` mode fails. The path violating global the time constraints. The root cause of the failure is also given in the verification log. It explains that the reason for the failure is: *Global time reach maximum value: 6. Need 7 time unit in total to complete time property* Prover highlights in red where the verification fails, in this example one more time unit is required to finish the process model.

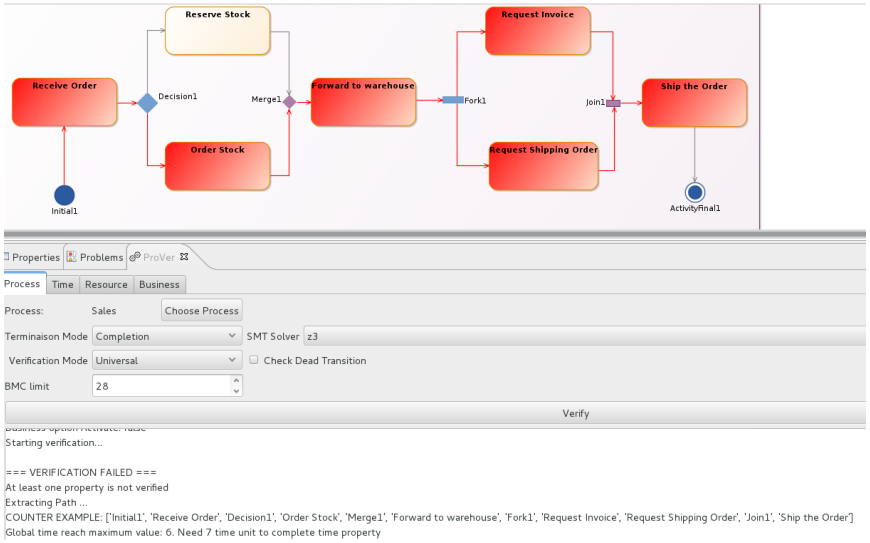

Figure 12: Verification Check Time Properties

But in `existential` mode verification succeeds. Effectively `Reserve Stock` activity is shorter than `Order Stock`, so it exist a path with 6 time unit. Prover highlights this path on green. (see figure 13)

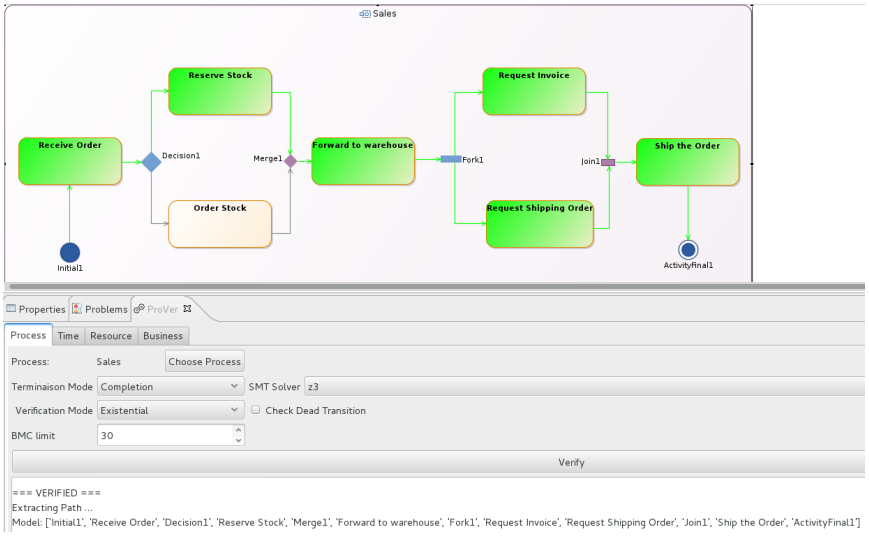

Figure 13: Verification Find Time Properties

7 Conclusion

Prover provides a set of properties that could be verified by anyone with no qualification on formal methods. The selection of all properties is entirely graphical and no special configuration is needed. All of this properties can be composed and verified all together. This tool is integrated in Merge platform and could be installed with an update site or locally with an archive file.

The use of ProVer provide an earlier detection of the flaws on the process models, which avoid bug, reduce the cost of production and a more safe product for the end user.

The set of properties implanted of the tool can be updated to allow a more specific verification the process models but this require some knowledge on formal methods.

References

[Christov 14] S.C. Christov, G. S. Avrunin , L.A. Clarke, American Medical Informatics Association Annual Symposium (AMIA 2014), November 15-17, 2014, Washington, DC, pp. 395-404. (UM-CS-2014-022)

[Mendling 07]]Jan Mendling, Gustaf Neumann, and Wil Van Der Aalst. Understanding the occurrence of errors in process models based on metrics.

In On the Move to Meaningful Internet Systems 2007 : CoopIS, DOA, ODBASE, GADA, and IS, pages 113–130. Springer, 2007. 3

[Vanhatalo 07]J.Vanhatalo, H. Völzer, and F.Leymann. Faster and more focused control-flow analysis for business process models through sese decomposition. In ICSOC 2007, pages 43–55. Springer, 2007. 3, 33, 143