

REPUBLIQUE DU CAMEROUN

Paix – Travail - Patrie

REPUBLIC OF CAMEROON

Peace – Work - Fatherland

UNIVERSITE DE DSCHANG

UNIVERSITY OF DSCHANG

Scholae Thesaurus Dschangensis Ibi Cordum



ECOLE DOCTORALE
POST GRADUATE SCHOOL

FACULTE DES SCIENCES
FACULTY OF SCIENCE

Département de Mathématiques et
Informatique

Department of Mathematics and Computer
Science

Réunion à distance de l'équipe FUCHSIA, Avril 2020

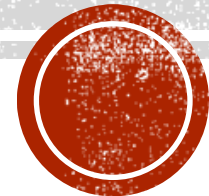
Utilisation de GEMOC Studio pour le développement d'une version simplifiée d'un langage de modélisation

Présenté par

ZEKENG NDADJI Milliam Maxime

Doctorant, Université de Dschang

ndadji.maxime@univ-dschang.org



PLAN DE LA PRÉSENTATION

- ❑ **L'ingénierie des langages avec l'approche GEMOC**
- ❑ **Travail Pratique: création des syntaxes abstraites et concrètes d'un langage d'arbre**
- ❑ **Perspectives et conclusion**

L'INGÉNIERIE DES LANGAGES AVEC L'APPROCHE GEMOC

GEMOC est une initiative libre et internationale, qui vise à coordonner et à diffuser les résultats de la recherche concernant le domaine de l'utilisation coordonnée de divers langages de modélisation.

("coordination" (globalization) des langages de modélisation).

GEMOC développe des techniques, des frameworks et des environnements pour faciliter la création, l'intégration et le traitement automatisé de langages de modélisation hétérogènes. Tous les résultats sont intégrés de manière transparente dans le logiciel GEMOC Studio.

- ❑ Site officiel de GEMOC → <http://gemoc.org>
- ❑ Télécharger GEMOC Studio → <http://gemoc.org/download.html>
- ❑ Documentation de GEMOC Studio → <https://archive.eclipse.org/gemoc/docs/milestones/3.1.0-20190627/>
- ❑ Trouvez une liste de publications relatives à GEMOC → <http://gemoc.org/publications.html>



**Language
Workbench**

Concevez et
composez vos
xDSML

<http://gemoc.org/studio>



**Modeling
Workbench**

Editez et déboguez vos
modèles hétérogènes



- ❑ Création et extension de méta-modèles avec des interpréteurs, des compilateurs et des vérificateurs de modèles (**Xtend/Kermeta**, **XMOF**, **Java**)
- ❑ Définition de la sémantique opérationnelle des langages exécutables ainsi qu'une couche d'animation au dessus de la syntaxe graphique concrète (**Sirius**)
- ❑ Déclaration, extension et assemblage des composants des langages exécutables (**Melange**)
- ❑ Génération automatique de méta-modèles de traces d'exécution pour prendre en charge les fonctionnalités avancées du modeling workbench
- ❑ Spécification d'un modèle de concurrence et de communication (**MoCC**) et son mapping à une syntaxe abstraite spécifique et aux fonctions d'exécution associées d'un langage de modélisation (**MoCCML**)
- ❑ Spécification des modèles de coordination des langages pour assurer la coordination automatique de l'exécution de modèles hétérogènes (**BCool**)

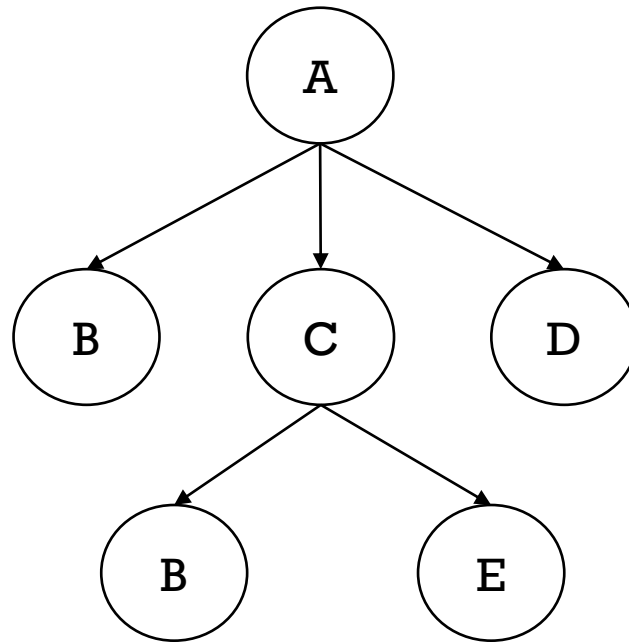


- Framework d'exécution générique composé de divers moteurs d'exécution et services d'exécution partagés
- Animation graphique et textuelle, définition des points d'arrêt et exécution du modèle étape par étape (simulation des modèles)
- Débogage avancé, navigation en avant et en arrière grâce aux traces d'exécution
- Analyse des contraintes de concurrence basée sur les traces d'exécution des modèles (analyse de la concurrence)
- Exécution simultanée et coordonnée de modèles hétérogènes
- Ajout dynamique de plugins au framework d'exécution

TRAVAIL PRATIQUE: CRÉATION DES SYNTAXES ABSTRAITES ET CONCRÈTES D'UN LANGAGE D'ARBRE

TRAVAIL PRATIQUE

UNE VERSION SIMPLIFIEE D'UN LANGAGE D'ARBRES (TREE_L)



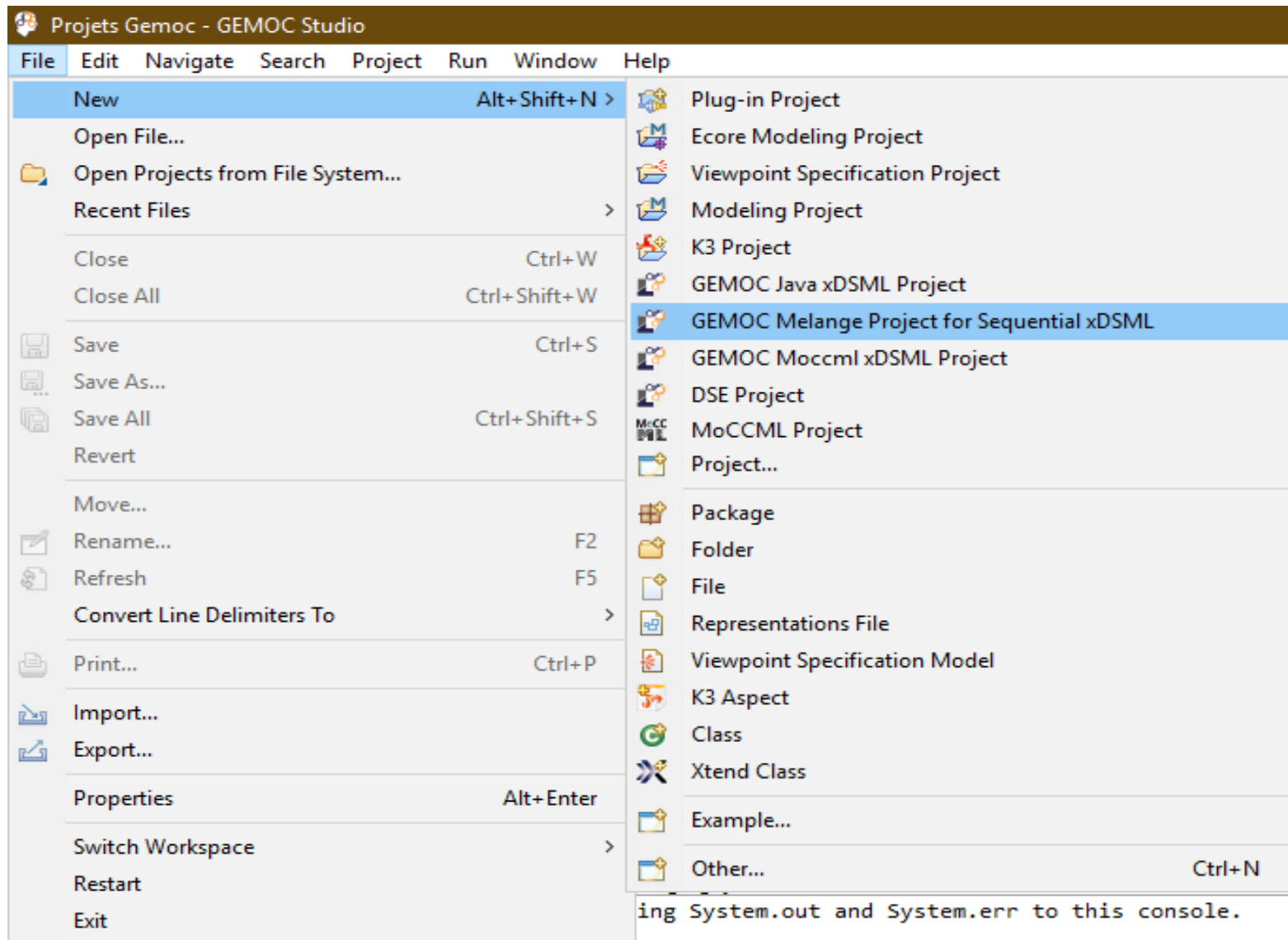
Un type Haskell pour ce langage:

```
data Artifact a = BuildArtifact { nodeLabel :: a, sons :: [Artifact a] }
```

1. Création du projet xDSML (executable Domain Specific Modeling Language)
2. Modélisation de la syntaxe abstraite du langage grâce à Ecore/EMF
3. Génération d'un éditeur (tree-based) pour le langage (Ecore/EMF)
4. Génération des interfaces Java du langage (Ecore/EMF)
5. Génération d'une syntaxe textuelle (xtext) concrète du langage
6. Génération d'une syntaxe graphique (sirius) concrète du langage
7. Utilisation du langage modélisé

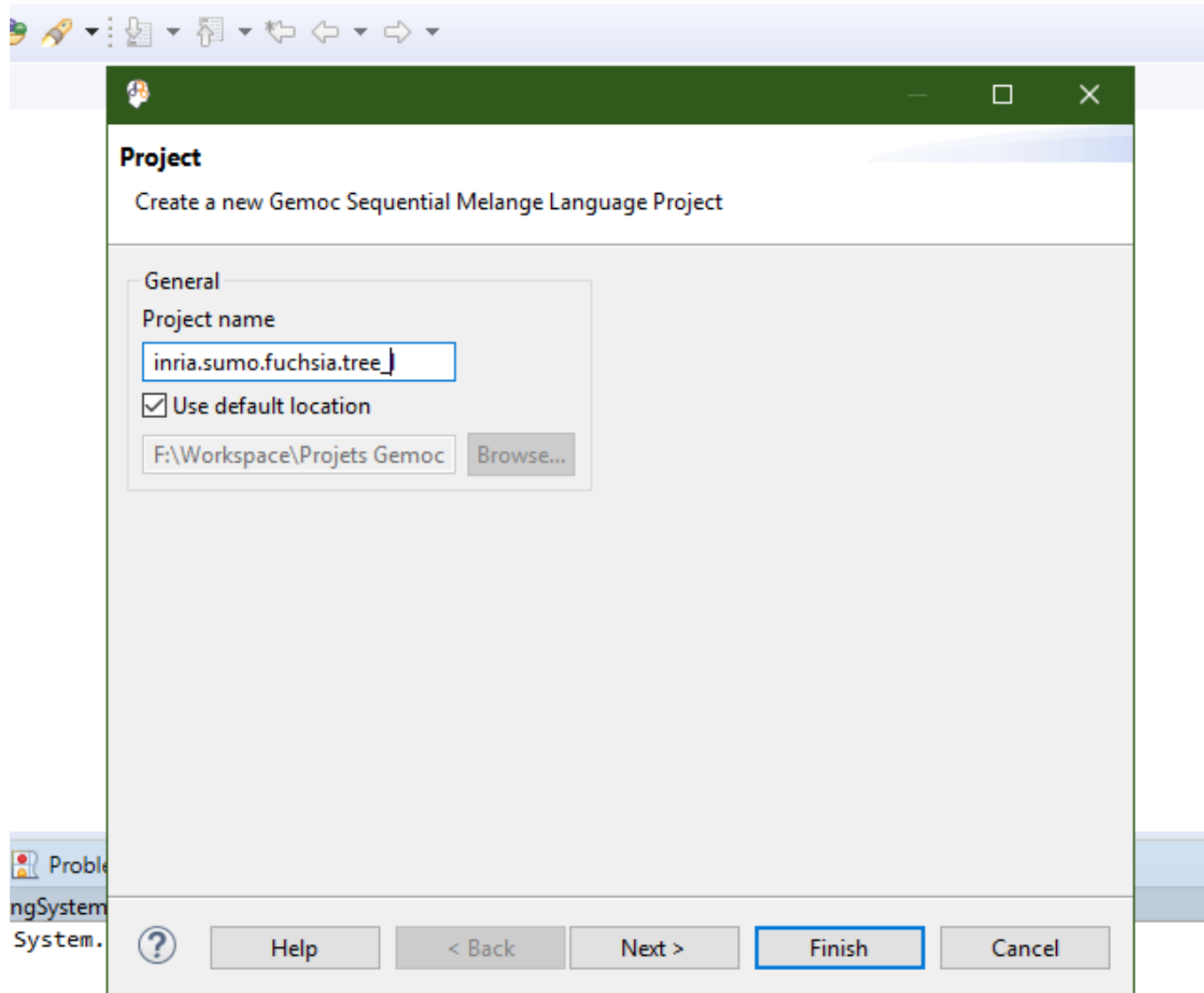
TRAVAIL PRATIQUE

1- CRÉATION DU PROJET XDSML



TRAVAIL PRATIQUE

1- CRÉATION DU PROJET XDSML



TRAVAIL PRATIQUE

1- CRÉATION DU PROJET XDSML

Projets Gemoc - inria.sumo.fuchsia.tree_l/src/inria/sumo/fuchsia/tree_l/Tree_l.melange - GEMOC Studio

File Edit Navigate Search Project Run Window Help

Project Explorer

- inria.sumo.fuchsia.tree_l
 - src
 - inria.sumo.fuchsia.tree_l
 - Tree_l.melange
 - JRE System Library [jre1.8.0_221]
 - Plug-in Dependencies
 - src-gen
 - META-INF
 - MANIFEST.MF
 - model-gen
 - build.properties
 - plugin.xml

```
package inria.sumo.fuchsia.tree_l

language Tree_l {
    // Melange generates an extended ecore in a language runtime project (inria.sumo.fuchsia.tree_l.tree_l)
    // that copies everything needed for the execution (abstract syntax and aspects),
    // in the SingleLanguage approach, the editor and animator must be written using this generated ecore a
    // syntax "platform:/resource/ecoreFilePath"

    // with qualified.class.name
}
```

Properties Problems Error Log @ Javadoc Console

Default MessagingSystem console
Redirecting System.out and System.err to this console.

TRAVAIL PRATIQUE

2- MODÉLISATION DE LA SYNTAXE ABSTRAITE DU LANGAGE GRÂCE À ECORE/EMF

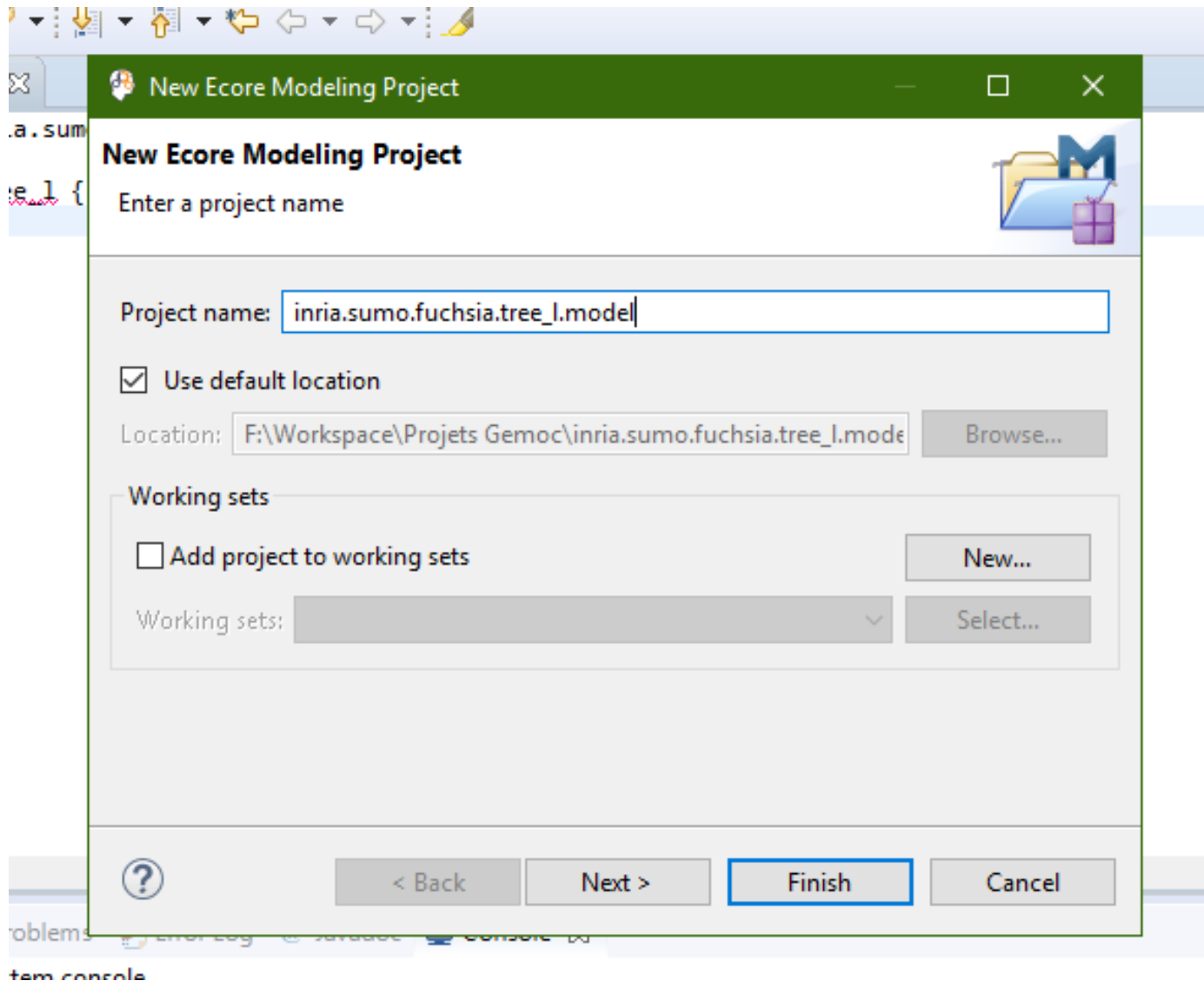
The screenshot displays the GEMOC Studio IDE interface. The title bar reads "Projets Gemoc - inria.sumo.fuchsia.tree_1/src/inria/sumo/fuchsia/tree_1/Tree_1.melange - GEMOC Studio". The menu bar includes "File", "Edit", "Navigate", "Search", "Project", "Run", "Window", and "Help". The toolbar contains various icons for file operations and development actions.

The Project Explorer on the left shows the project structure:

- inria.sumo.fuchsia.tree_1
 - src
 - inria.sumo.fuchsia.tree_1
 - Tree_1.melange
 - JRE System Library [jre1.8.0_221]
 - Plug-in Dependencies
 - src-gen
 - META-INF
 - MANIFEST.MF
 - model-gen
 - build.properties
 - plugin.xml

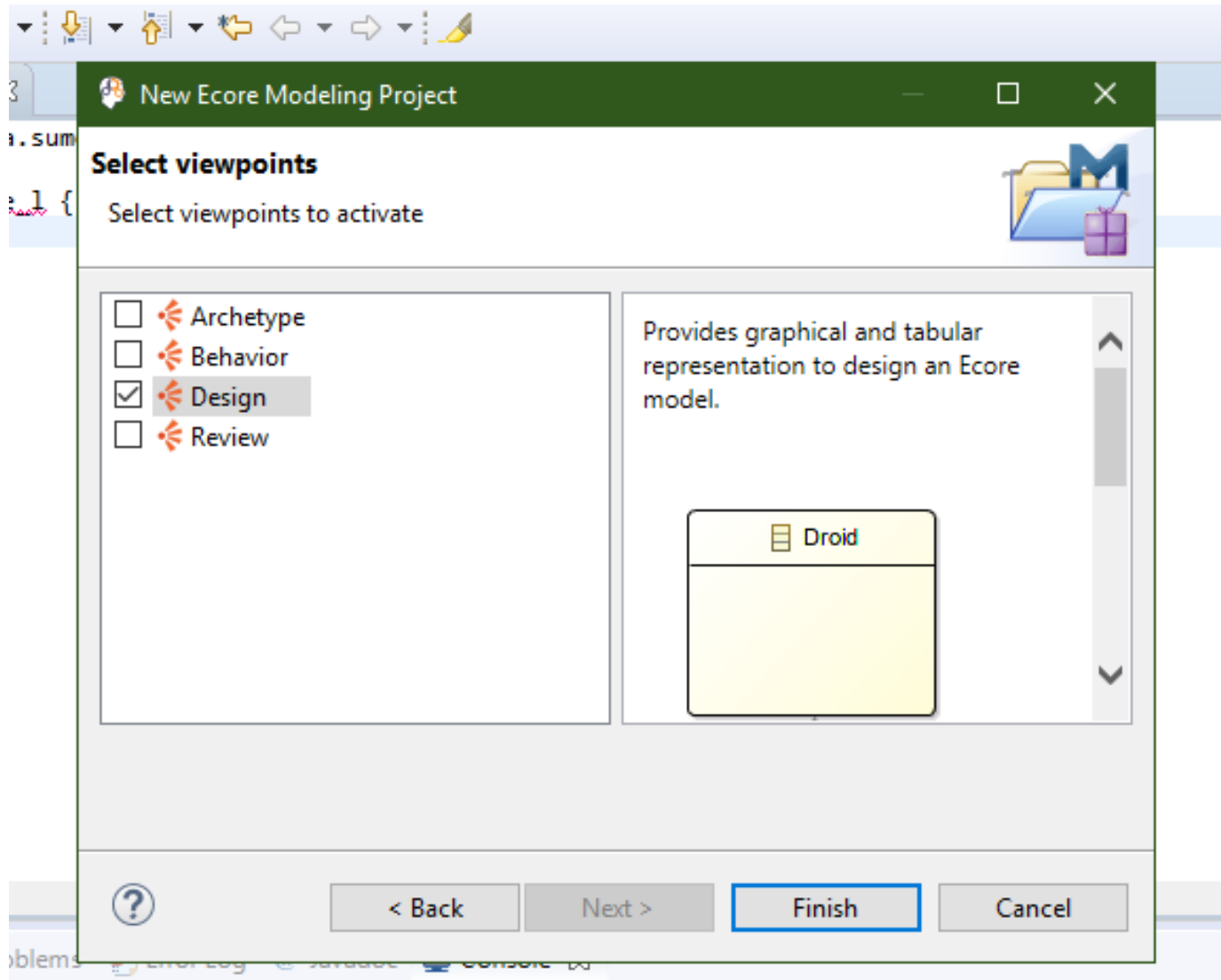
TRAVAIL PRATIQUE

2- MODÉLISATION DE LA SYNTAXE ABSTRAITE DU LANGAGE GRÂCE À ECORE/EMF



TRAVAIL PRATIQUE

2- MODÉLISATION DE LA SYNTAXE ABSTRAITE DU LANGAGE GRÂCE À Ecore/EMF



em console

TRAVAIL PRATIQUE

2- MODÉLISATION DE LA SYNTAXE ABSTRAITE DU LANGAGE GRÂCE À ECORE/EMF

The screenshot displays the GEMOC Studio IDE interface. The title bar shows the project path: `Projets Gemoc - inria.sumo.fuchsia.tree_1/src/inria/sumo/fuchsia/tree_1/Tree_1.melange - GEMOC Studio`. The menu bar includes `File Edit Navigate Search Project Run Window Help`. The toolbar contains various icons for file operations and development actions. The **Model Explorer** on the left shows a tree structure for the project `inria.sumo.fuchsia.tree_1`, with the `model.ecore` file selected under the `model` package. The main editor window shows the content of `model.ecore`, with the following line highlighted and circled in red:

```
syntax "platform:/resource/inria.sumo.fuchsia.tree_1.model/model/model.ecore"
```

The **Properties** panel at the bottom indicates "No properties available". The Windows taskbar at the bottom shows the system tray with the date `14/04/2020` and time `07:52`.

TRAVAIL PRATIQUE

2- MODÉLISATION DE LA SYNTAXE ABSTRAITE DU LANGAGE GRÂCE À ECORE/EMF

The screenshot displays the GEMOC Studio interface for modeling an abstract language syntax. The main workspace shows a class diagram for the 'model' package. A class named 'Artifact' is defined with an attribute 'nodeLabel : EString'. A self-referencing association is shown on the 'Artifact' class, labeled 'sons' with a multiplicity of '[0..*]'. Red arrows point to the class name 'Artifact', the attribute 'nodeLabel : EString', and the association line, with labels 'Classe', 'Attribut', and 'Référence' respectively.

The left sidebar shows the Model Explorer with the project structure. The 'model' package is selected, showing its contents: 'Artifact' class, 'nodeLabel : EString' attribute, and 'sons : Artifact' association.

The bottom right shows the Properties view for the 'model' package, with the following details:

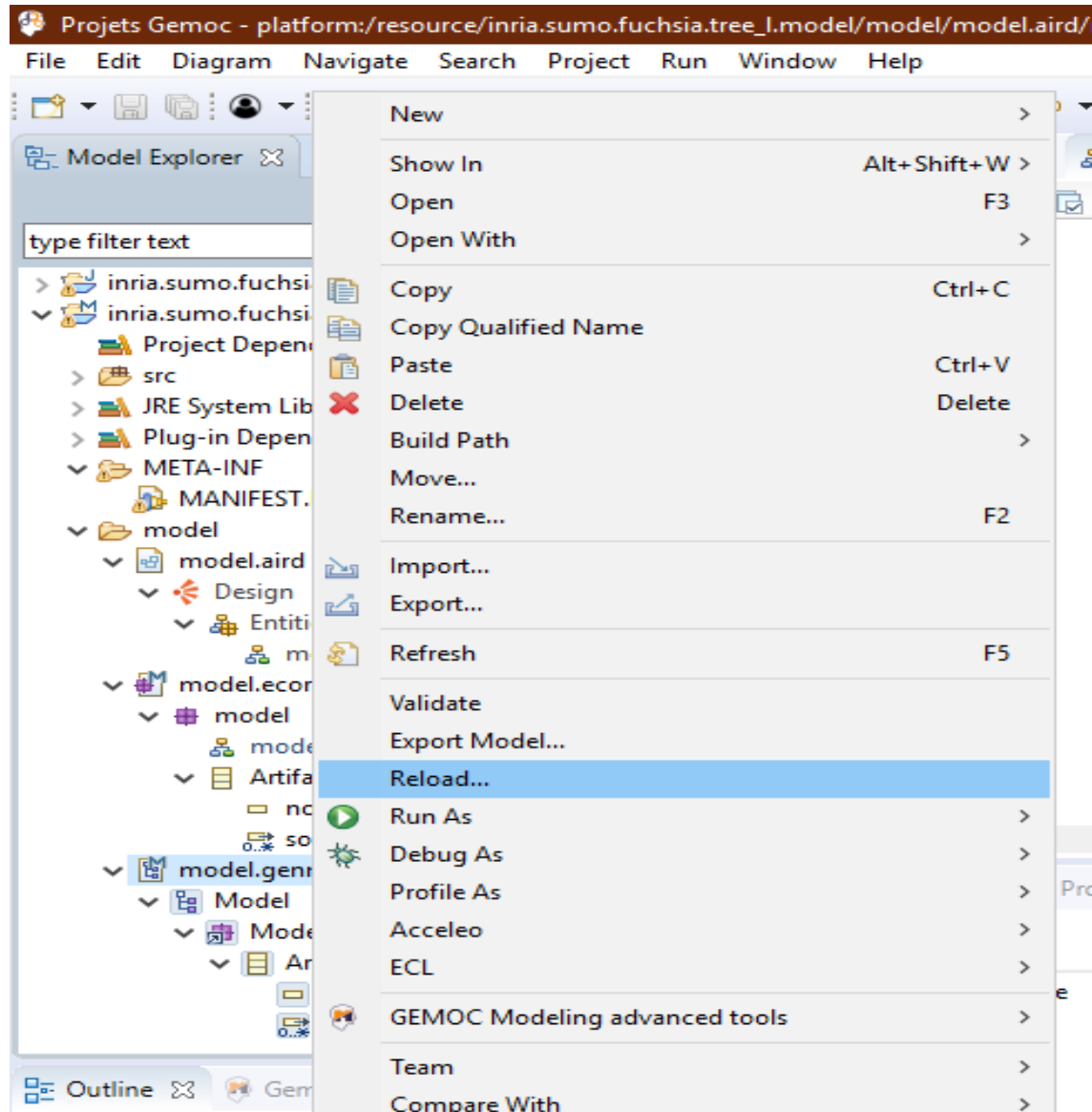
- Name: model
- Ns URI: http://www.example.org/model
- Ns Prefix: model

The right sidebar shows the Palette with various modeling elements like Class, Feature, Relation, Reference, etc.

En savoir plus sur EMF: <https://www.vogella.com/tutorials/EclipseEMF/article.html>
<https://www.eclipse.org/sirius/doc/user/general/Modeling%20Project.html>

TRAVAIL PRATIQUE

2- MODÉLISATION DE LA SYNTAXE ABSTRAITE DU LANGAGE GRÂCE À ECORE/EMF



TRAVAIL PRATIQUE

3- GÉNÉRATION D'UN ÉDITEUR (TREE-BASED) POUR LE LANGAGE (ECORE/EMF)

The screenshot displays the GEMOC Studio interface. The title bar reads "Projets Gemoc - inria.sumo.fuchsia.tree_l.model/model/model.genmodel - GEMOC Studio". The menu bar includes "File", "Edit", "Navigate", "Search", "Project", "Generator", "Run", "Window", and "Help". The toolbar contains various icons for file operations and execution. The "Model Explorer" on the left shows a tree structure of the project, with the "model.genmodel" file selected and highlighted by a red rectangle. The "Properties" window at the bottom shows the "Artifact" property for the selected file. The context menu is open over the "model.genmodel" file, listing several options: "Generate Model Code", "Generate Edit Code", "Generate Editor Code", "Generate Test Code", "Generate All" (highlighted in blue), "Open Ecore", "Open GenModel", "Undo" (Ctrl+Z), "Redo Reconcile" (Ctrl+Y), "Cut", "Copy", "Paste", "Delete", "Find/Replace...", "Live Validation" (checked), "Run As", "Debug As", "Profile As", "Validate", "Team", "Compare With", "Replace With", "Show EClass information", "Show References", "OCL", and "Refresh".

TRAVAIL PRATIQUE

3- GÉNÉRATION D'UN ÉDITEUR (TREE-BASED) POUR LE LANGAGE (ECORE/EMF)

The screenshot shows the GEMOC Studio IDE with the following components:

- Project Explorer (Left):** Displays the project structure for 'inria.sumo.fuchsia.tree_l'. The 'src-gen' folder is expanded, showing the package 'inria.sumo.fuchsia.tree_l.model' and its sub-packages 'inria.sumo.fuchsia.tree_l.model.impl' and 'inria.sumo.fuchsia.tree_l.model.util'. The 'Artifact.java' file is highlighted.
- Code Editor (Center):** Shows the source code for 'Artifact.java'. The code includes package declarations, imports, and a public interface definition for 'Artifact' that extends 'EObject'. The interface has a method 'getNodeLabel()' with a Javadoc comment.
- Properties/Problems/Console (Bottom):** The Properties panel is active, showing a table with 'Property' and 'Value' columns. The Problems and Console panels are empty.

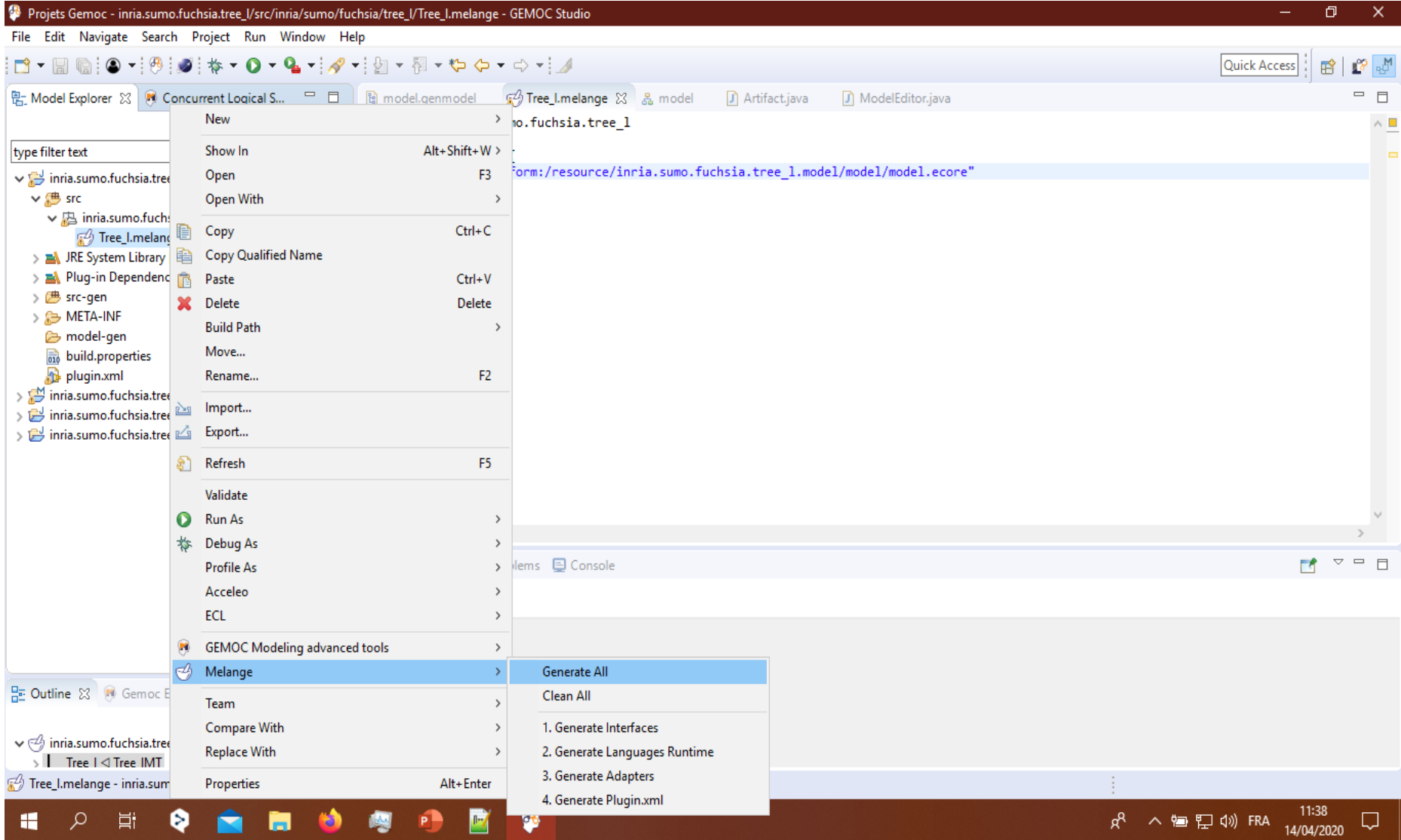
```
package inria.sumo.fuchsia.tree_l.model;

import org.eclipse.emf.common.util.EList;

/**
 * @! -- begin-user-doc --
 * A representation of the model object '<em><b>Artifact</b></em>'.
 * @! -- end-user-doc --
 *
 * <p>
 * The following features are supported:
 * </p>
 * <ul>
 * <li>{@link inria.sumo.fuchsia.tree_l.model.Artifact#getNodeLabel <em>Node Label</em>}</li>
 * <li>{@link inria.sumo.fuchsia.tree_l.model.Artifact#getSons <em>Sons</em>}</li>
 * </ul>
 *
 * @see inria.sumo.fuchsia.tree_l.model.ModelPackage#getArtifact()
 * @model
 * @generated
 */
public interface Artifact extends EObject {
    /**
     * Returns the value of the '<em><b>Node Label</b></em>' attribute.
     */
}
```

TRAVAIL PRATIQUE

4- GÉNÉRATION DES INTERFACES JAVA DU LANGAGE (ECORE/EMF)



TRAVAIL PRATIQUE

4- GÉNÉRATION DES INTERFACES JAVA DU LANGAGE (ECORE/EMF)

The screenshot displays the GEMOC Studio IDE interface. The top menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and development tools. The Model Explorer on the left shows a project structure with a red circle highlighting the path: inria.sumo.fuchsia.tree_l.tree_l.model.Artifact.java. The main editor window shows the generated Java code for the Artifact interface, which extendsEObject and includes methods like getNodeLabel() and setNodeLabel(). The Properties, Problems, and Console panels are visible at the bottom, and the status bar shows 'Writable', 'Smart Insert', and '1:1'.

```
@see inria.sumo.fuchsia.tree_l.tree_l.model.ModelPackage#getArtifact()
* @model
* @generated
*/
public interface Artifact extends EObject {
    /**
     * Returns the value of the '<em><b>Node Label</b></em>' attribute.
     * <!-- begin-user-doc -->
     * <p>
     * If the meaning of the '<em>Node Label</em>' attribute isn't clear,
     * there really should be more of a description here...
     * </p>
     * <!-- end-user-doc -->
     * @return the value of the '<em>Node Label</em>' attribute.
     * @see #setNodeLabel(String)
     * @see inria.sumo.fuchsia.tree_l.tree_l.model.ModelPackage#getArtifact_NodeLabel()
     * @model
     * @generated
     */
    String getNodeLabel();

    /**
     * Sets the value of the '{@link inria.sumo.fuchsia.tree_l.tree_l.model.Artifact#getNodeLabel <em>Node Label</em>}' attribute.
     * <!-- begin-user-doc -->
     * <!-- end-user-doc -->
     */
}
```


TRAVAIL PRATIQUE

5- GÉNÉRATION D'UNE SYNTAXE TEXTUELLE (XTEXT) CONCRÈTE DU LANGAGE

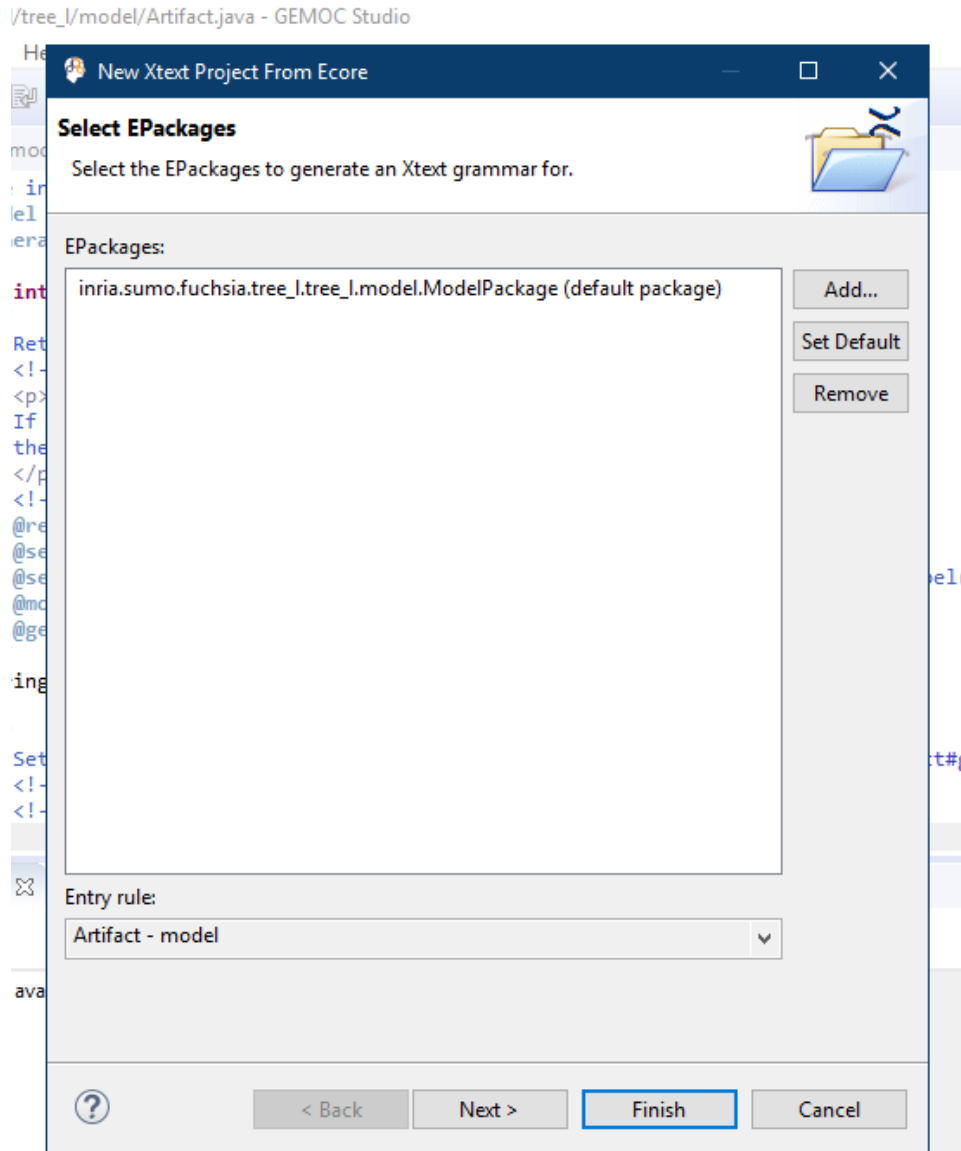
The screenshot shows the GEMOC Studio IDE interface. The main window displays the source code for `Artifact.java` in the `inria.sumo.fuchsia.tree_1.model` package. The code includes a class `Artifact` that extends `EObject` and has a method `getNodeLabel()`. The IDE's menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The Model Explorer on the left shows the project structure, with `inria.sumo.fuchsia.tree_1` selected. The Run menu is open, and the `GEMOC Language` option is selected, which has opened a sub-menu with the following options:

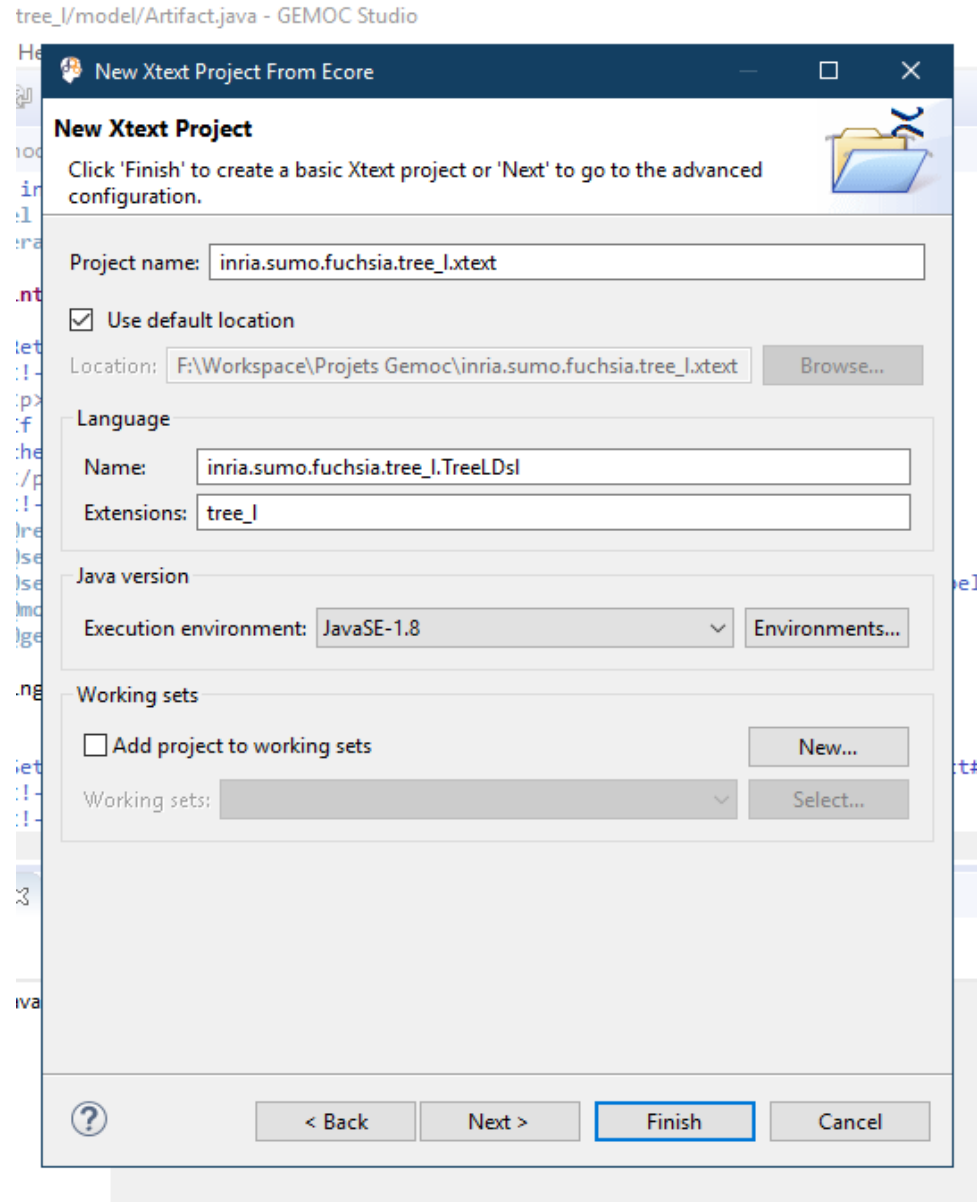
- Generate Multidimensional Trace Addon project for language
- Create Domain Model Project for language
- Create Sirius Editor Project for language
- Create XText Editor Project for language
- Create Animator Project for language

The `Create XText Editor Project for language` option is highlighted in blue. The Windows taskbar at the bottom shows the system tray with the date and time: 11:51, 14/04/2020.

TRAVAIL PRATIQUE

5- GÉNÉRATION D'UNE SYNTAXE TEXTUELLE (XTEXT) CONCRÈTE DU LANGAGE





TRAVAIL PRATIQUE

5- GÉNÉRATION D'UNE SYNTAXE TEXTUELLE (XTEXT) CONCRÈTE DU LANGAGE

The screenshot shows the GEMOC Studio interface. The left sidebar displays a project tree for 'inia.sumo.fuchsia.tree_l'. The main editor window shows the Xtext grammar code for 'Artifact' and 'EString'. A red box highlights the code, and a red arrow points to it with the text 'Grammaire générée'.

```
// automatically generated by Xtext
grammar inia.sumo.fuchsia.tree_l.TreeLdsl with org.eclipse.xtext.common.Terminals

import "http://inia.sumo.fuchsia.tree_l.tree_l/model/"
import "http://www.eclipse.org/emf/2002/Ecore" as ecore

Artifact returns Artifact:
  {Artifact}
  'Artifact'
  '{'
    ('nodeLabel' nodeLabel=EString)?
    ('sons' '(' sons+=[Artifact|EString] ( "," sons+=[Artifact|EString])* ')' )?
  '>';

EString returns ecore::EString:
  STRING | ID;
```

TRAVAIL PRATIQUE

5- GÉNÉRATION D'UNE SYNTAXE TEXTUELLE (XTEXT) CONCRÈTE DU LANGAGE

The screenshot shows the Eclipse IDE interface with the following components:

- Model Explorer:** Displays the project structure for 'inria.sumo.fuchsia.tree_l'. The file 'inria.sumo.fuchsia.tree_l.xtext' is selected.
- Context Menu:** A right-click menu is open over the selected file, showing options: '1 Generate TreeLDsl (tree_l) Language Infrastructure', '2 Launch Runtime Eclipse', '3 Test-Gemoc-Language', '4 GenerateGagDSL.mwe2', and '5 New_configuration'. The 'Run As' submenu is also visible.
- Editor:** Displays the content of 'TreeLDsl.xtext'. The visible code includes:

```
Artifact.java TreeLDsl with org.eclipse.xtext.common.Terminals
inria.sumo.fuchsia.tree_l.model/
/emf/2002/Ecore" as ecore

('sons' '(' sons+=[Artifact|EString] ( "," sons+=[Artifact|EString])* ')')?
};

EString returns ecore::EString:
STRING | ID;
```
- Properties:** Shows 'No properties available'.
- Outline:** Shows the 'EString' element.
- Taskbar:** Shows the Windows taskbar with the date '14/04/2020' and time '12:05'.

TRAVAIL PRATIQUE

5- GÉNÉRATION D'UNE SYNTAXE TEXTUELLE (XTEXT) CONCRÈTE DU LANGAGE

The screenshot shows the Eclipse IDE interface for a project named 'Projets Gemoc'. The main editor displays the source code for the file 'TreeLdslFormatter.xtend'. The code is as follows:

```
* generated by Xtext 2.14.0
package inria.sumo.fuchsia.formatting2

import com.google.inject.Inject
import inria.sumo.fuchsia.services.TreeLdslGrammarAccess
import org.eclipse.xtext.formatting2.AbstractFormatter2
import org.eclipse.xtext.formatting2.IFormattableDocument

class TreeLdslFormatter extends AbstractFormatter2 {

    @Inject extension TreeLdslGrammarAccess

    override format(Object obj, IFormattableDocument document) {
        |
    }

    // TODO: implement for
}
```

The 'override format' method is highlighted with a red box. The Outline view on the right shows the project structure, including the 'TreeLdslFormatter' class and its 'format' method. The Console view at the bottom shows the output of the Xtext generation process:

```
<terminated> Generate TreeLdsl (tree_l) Language Infrastructure [Mwe2 Launch] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (14 avr. 2020 à 12:29:55)
1674 [main] INFO   clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/xbase/Xtype' from 'platform:/resource/org.ecl
1752 [main] INFO   clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/xbase/Xbase' from 'platform:/resource/org.ecl
1752 [main] INFO   clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/common/JavaVMTypes' from 'platform:/resource/
1768 [main] INFO   clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://inria.sumo.fuchsia.tree_l.tree_l/model/' from 'platform:/resource/ir
3784 [main] INFO   erator.parser.antlr.AntlrToolFacade - downloading file from 'http://download.itemis.com/antlr-generator-3.2.0-patch.jar' ...
114201 [main] INFO erator.parser.antlr.AntlrToolFacade - finished downloading.
115983 [main] INFO text.xtext.generator.XtextGenerator - Generating inria.sumo.fuchsia.TreeLdsl
134023 [main] INFO text.xtext.generator.XtextGenerator - Generating common infrastructure
134242 [main] INFO .emf.mwe2.runtime.workflow.Workflow - Done.
```

TRAVAIL PRATIQUE

6- GÉNÉRATION D'UNE SYNTAXE GRAPHIQUE (SIRIUS) CONCRÈTE DU LANGAGE

The screenshot displays the GEMOC Studio IDE interface. The 'GEMOC Language' menu is open, showing several options for generating projects and editors. The 'Create Sirius Editor Project for language' option is selected. The background shows the Eclipse IDE with a project explorer on the left, a central editor window displaying code, and a console window at the bottom.

Project Explorer:

- inria.sumo.fuchsia.tree_l
- inria.sumo.fuchsia.tree_l.model
- inria.sumo.fuchsia.tree_l.model.edi
- inria.sumo.fuchsia.tree_l.tree_l
- inria.sumo.fuchsia.tree_l.xtext
- inria.sumo.fuchsia.tree_l.xtext.ide
- inria.sumo.fuchsia.tree_l.xtext.tests
- inria.sumo.fuchsia.tree_l.xtext.ui
- inria.sumo.fuchsia.tree_l.xtext.ui.te

Menu Items:

- New
- Go Into
- Show In (Alt+Shift+W)
- Copy (Ctrl+C)
- Copy Qualified Name
- Paste (Ctrl+V)
- Delete (Delete)
- Build Path
- Refactor (Alt+Shift+T)
- Import...
- Export...
- Refresh (F5)
- Close Project
- Close Unrelated Project
- Validate
- Run As
- Debug As
- Profile As
- Restore from Local History...
- Accelero
- GEMOC Language**
- GEMOC Java xDSML
- Team
- Compare With
- Plug-in Tools
- Configure
- Source
- Properties (Alt+Enter)

Editor Content:

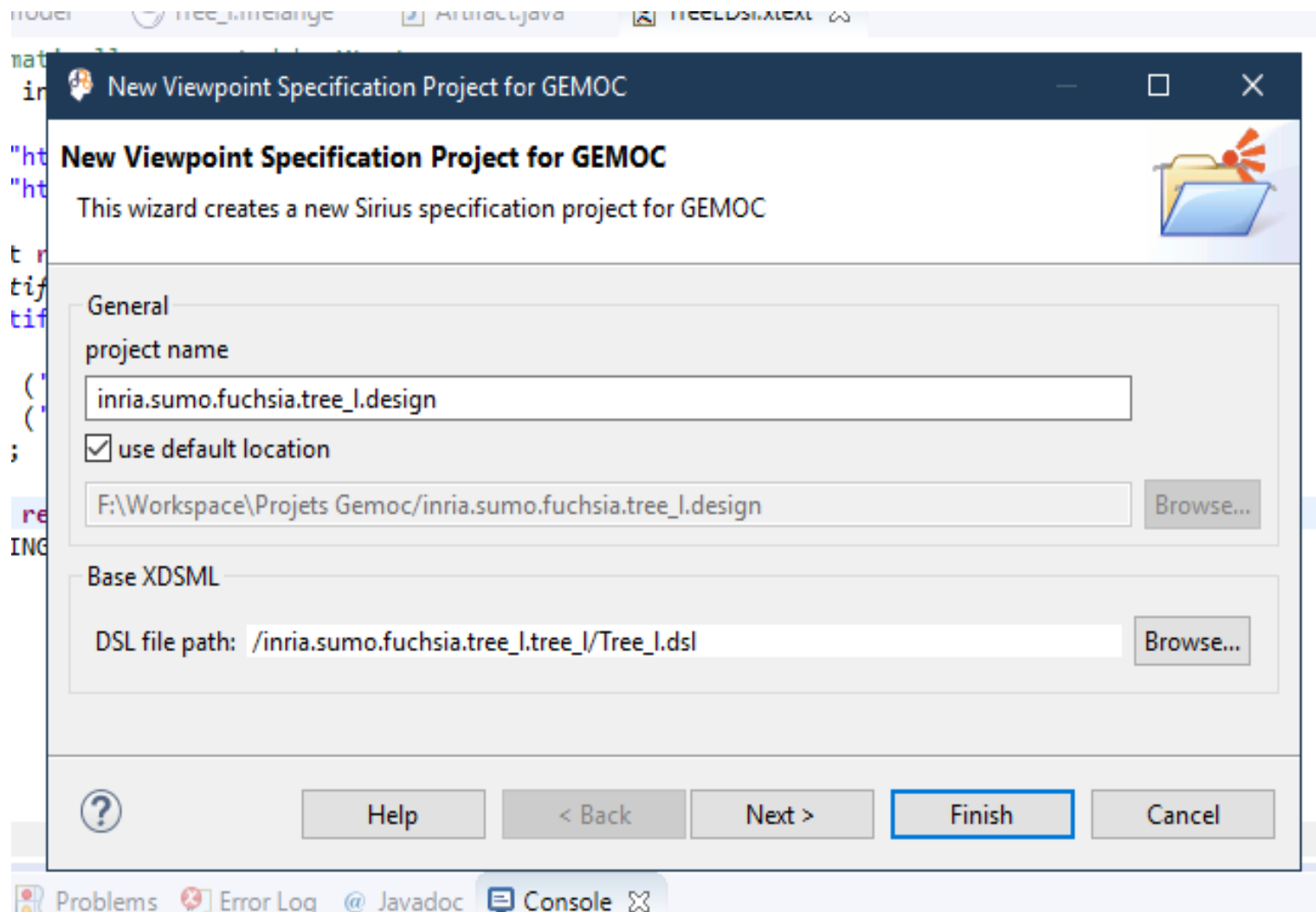
```
Artifact.java TreeLDsL.txt  
1.TreeLDsL with org.eclipse.xtext.common.Terminals  
a.tree_l.tree_l/model/  
mf/2002/Ecore" as.ecore  
String)?  
act[EString] ( "," sons+=[Artifact[EString])* ']' )?  
Javadoc Console  
e [Mwe2 Launch] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (14 avr. 2020 à 12:05:32)  
ia.tree_l.xtext.ui at 'file:/F:/Workspace/Projets%20Gemoc/ir  
ia.tree_l.xtext.ui.tests at 'file:/F:/Workspace/Projets%20G  
istered Packages will not be registered in the global EPacka  
pse.org/Xtext/Xbase/XAnnotations' from 'platform:/resource/c  
pse.org/xtext/xbase/Xtype' from 'platform:/resource/org.ecl  
pse.org/xtext/xbase/Xbase' from 'platform:/resource/org.ecl  
pse.org/xtext/common/JavaVMTypes' from 'platform:/resource/c  
mo.fuchsia.tree_l.tree_l/model/' from 'platform:/resource/ir  
.AntlrToolFacade - downloading file from 'http://download.itemis.com/antlr-generator-3.2.0-patch.jar' ...
```

Outline:

- grammar inria.sumo.fuchsia.tree_l.TreeLDsL
 - import model
 - import.ecore as.ecore
 - Artifact
 - EString

TRAVAIL PRATIQUE

6- GÉNÉRATION D'UNE SYNTAXE GRAPHIQUE (SIRIUS) CONCRÈTE DU LANGAGE



TRAVAIL PRATIQUE

6- GÉNÉRATION D'UNE SYNTAXE GRAPHIQUE (SIRIUS) CONCRÈTE DU LANGAGE

The screenshot displays the GEMOC Studio IDE interface. The top menu bar includes File, Edit, Navigate, Search, Project, Viewpoint Specification Editor, Run, Window, and Help. The Project Explorer on the left shows a project structure with a red circle highlighting the 'tree_l.design' folder. The Sirius Specification Editor in the center shows a tree view with 'MyViewpoint' containing 'inria.sumo.fuchsia.tree_l.design.Services'. The Console window at the bottom shows logs for 'Generate TreeDsl (tree_l) Language Infrastructure [Mwe2 Launch]'. The logs indicate the registration of various GenModel instances and the download of the 'antlr-generator-3.2.0-patch.jar' file.

```
Generate TreeDsl (tree_l) Language Infrastructure [Mwe2 Launch] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (14 avr. 2020 à 12:05:32)
847 [main] INFO lipse.emf.mwe.utils.StandaloneSetup - Registering project inria.sumo.fuchsia.tree_l.xtext.ui at 'file:/F:/Workspace/Projets%20Gemoc/ir
847 [main] INFO lipse.emf.mwe.utils.StandaloneSetup - Registering project inria.sumo.fuchsia.tree_l.xtext.ui.tests at 'file:/F:/Workspace/Projets%20Ge
863 [main] INFO lipse.emf.mwe.utils.StandaloneSetup - Using resourceSet registry. The registered Packages will not be registered in the global EPacka
1769 [main] INFO clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/Xtext/Xbase/XAnnotations' from 'platform:/resource/c
1784 [main] INFO clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/xbase/Xtype' from 'platform:/resource/org.ecl
1831 [main] INFO clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/xbase/Xbase' from 'platform:/resource/org.ecl
1831 [main] INFO clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/common/JavaVMTypes' from 'platform:/resource/c
1847 [main] INFO clipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://inria.sumo.fuchsia.tree_l.tree_l/model/' from 'platform:/resource/i
4534 [main] INFO erator.parser.antlr.AntlrToolFacade - downloading file from 'http://download.itemis.com/antlr-generator-3.2.0-patch.jar' ...
```

TRAVAIL PRATIQUE

6- GÉNÉRATION D'UNE SYNTAXE GRAPHIQUE (SIRIUS) CONCRÈTE DU LANGAGE

The screenshot displays the GEMOC Studio IDE interface. The Project Explorer on the left shows the project structure for 'inria.sumo.fuchsia.tree_l', with 'Tree_L.dsl' highlighted. The main editor shows the DSL configuration for 'Tree_L.dsl', with the 'sirius' property highlighted in red, pointing to the description file: `sirius = platform:/resource/inria.sumo.fuchsia.tree_l.design/description/tree_l.odesign`. The Console at the bottom shows the execution output of the 'Generate TreeLdsl' command, including the following log entries:

```
Generate TreeLdsl (tree_l) Language Infrastructure [Mwe2 Launch] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (14 avr. 2020 à 12:05:32)
847 [main] INFO lipse.emf.mwe.utils.StandaloneSetup - Registering project inria.sumo.fuchsia.tree_l.xtext.ui at 'file:/F:/Workspace/Projets%20Gemoc/ir
847 [main] INFO lipse.emf.mwe.utils.StandaloneSetup - Registering project inria.sumo.fuchsia.tree_l.xtext.ui.tests at 'file:/F:/Workspace/Projets%20G
863 [main] INFO lipse.emf.mwe.utils.StandaloneSetup - Using resourceSet registry. The registered Packages will not be registered in the global EPacka
1769 [main] INFO eclipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/Xtext/Xbase/XAnnotations' from 'platform:/resource/
1784 [main] INFO eclipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/xbase/Xtype' from 'platform:/resource/org.ecl
1831 [main] INFO eclipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/xbase/Xbase' from 'platform:/resource/org.ecl
1831 [main] INFO eclipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xtext/common/JavaVMTypes' from 'platform:/resource/
1847 [main] INFO eclipse.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://inria.sumo.fuchsia.tree_l.tree_l/model/' from 'platform:/resource/i
4534 [main] INFO erator.parserantlr.AntlrToolFacade - downloading file from 'http://download.itemis.com/antlr-generator-3.2.0-patch.jar' ...
```

TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ

The screenshot displays the GEMOC Studio IDE interface. The main editor shows a Java class named `TreeDslFormatter` with the following code:

```
@Inject extension TreeDslGrammarAccess
override format(Object obj, IFormattableDocument document) {
}
// TODO: implement for
```

The Project Explorer on the left shows the project structure for `inria.sumo.fuchsia.tree_1`. The Outline view on the right shows the class hierarchy, including `TreeDslGrammarAccess` and `format(Object, IFormattableDocument)`. The Console window at the bottom shows the output of the `Generate TreeDsl (tree_1) Language Infrastructure` task, including the following log entries:

```
<terminated> Generate TreeDsl (tree_1) Language Infrastructure [Mwe2 Launch] C:\Program Files\Java\jre1.8.0_221\bin\javaw.exe (14 avr. 2020 à 12:29:55)
1674 [main] INFO clipe.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xttext/xbase/Xtype' from 'platform:/resource/org.ecl:
1752 [main] INFO clipe.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xttext/xbase/Xbase' from 'platform:/resource/org.ecl:
1752 [main] INFO clipe.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://www.eclipse.org/xttext/common/JavaVMTypes' from 'platform:/resource/c:
1768 [main] INFO clipe.emf.mwe.utils.GenModelHelper - Registered GenModel 'http://inria.sumo.fuchsia.tree_1.tree_1/model/' from 'platform:/resource/ir
3784 [main] INFO erator.parser.antlr.AntlrToolFacade - downloading file from 'http://download.itemis.com/antlr-generator-3.2.0-patch.jar' ...
114201 [main] INFO erator.parser.antlr.AntlrToolFacade - finished downloading.
115983 [main] INFO text.xtext.generator.XtextGenerator - Generating inria.sumo.fuchsia.TreeDsl
134023 [main] INFO text.xtext.generator.XtextGenerator - Generating common infrastructure
134242 [main] INFO .emf.mwe2.runtime.workflow.Workflow - Done.
```

TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ

The screenshot displays the Eclipse IDE interface with the 'Run Configurations' dialog box open. The dialog is titled 'Create, manage, and run configurations' and is used to launch an Eclipse application. The configuration name is 'Test-Gemoc-Language'. The 'Workspace Data' section shows the location as '\${workspace_loc}/../runtime-Test-Gemoc-Language'. The 'Program to Run' section is set to 'Run a product' with the value 'org.eclipse.gemoc.gemoc_studio.branding.gemoc_studio'. The 'Java Runtime Environment' section is set to 'Default (javaw)' and 'Execution environment: JavaSE-1.8 (jre1.8.0_221)'. The 'Bootstrap entries' field is empty. The 'Run' button is highlighted in blue. In the background, the Project Explorer shows a tree structure of files, and the Console shows a log message: '134242 [main] INFO .emf.mwe2.runtime.workflow.Workflow - Done.' The system tray at the bottom right shows the date and time: '12:47 14/04/2020'.

TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ

The screenshot displays the GEMOC Studio IDE interface. A red box highlights the window title bar: `runtime-Test-Gemoc-Language - GEMOC Studio`. The main window shows the 'New Project' wizard dialog box with the following content:

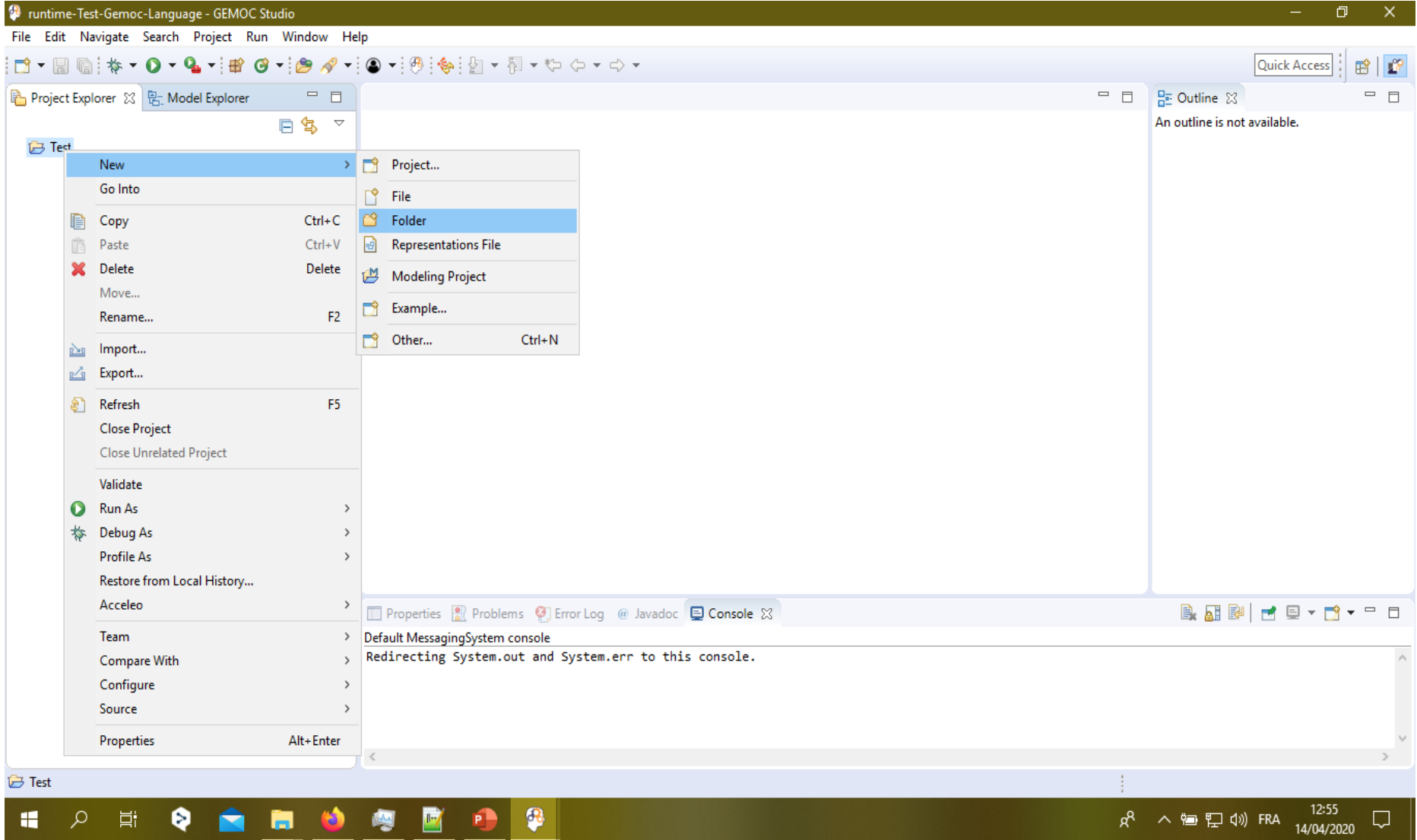
- Select a wizard**
Create a new project resource
- Wizards:**
type filter text
- General** (expanded)
 - Project** (selected and highlighted with a red box)
 - Acceleo Model to Text
 - AspectJ
 - CCSL
 - Eclipse Modeling Framework
 - GEMOC Language
 - Java
 - JavaFX
 - Kermeta 3
 - Maven
 - Melange

At the bottom of the dialog, there are buttons for '< Back', 'Next >', 'Finish', and 'Cancel'. The 'Next >' button is highlighted with a blue border.

The background IDE interface includes the Project Explorer and Model Explorer on the left, and the Outline window on the right showing 'An outline is not available.' The bottom status bar shows the system tray with the date and time: 14/04/2020, 12:54.

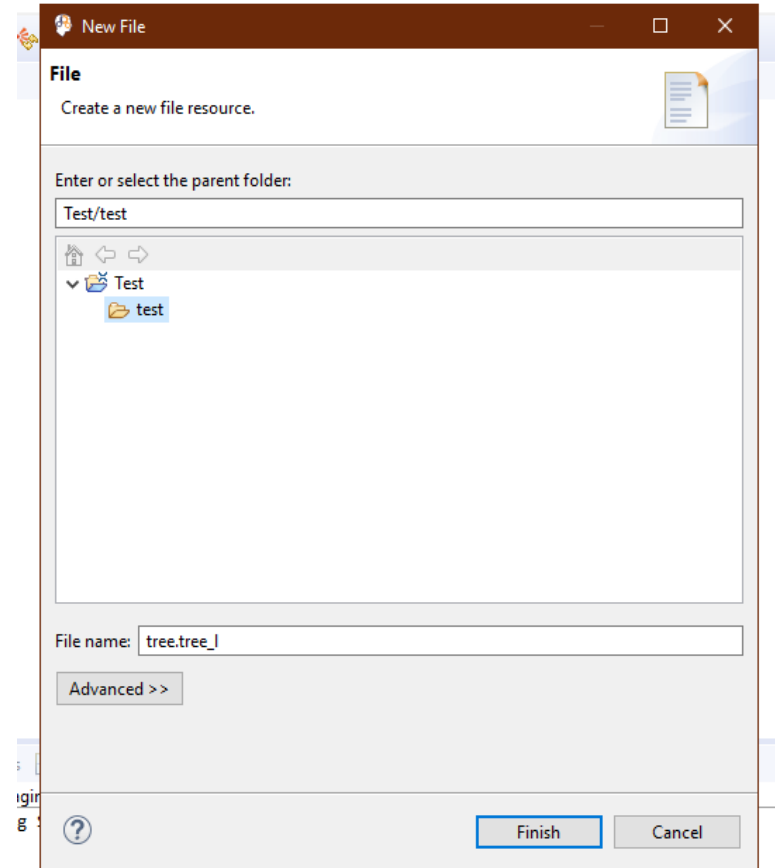
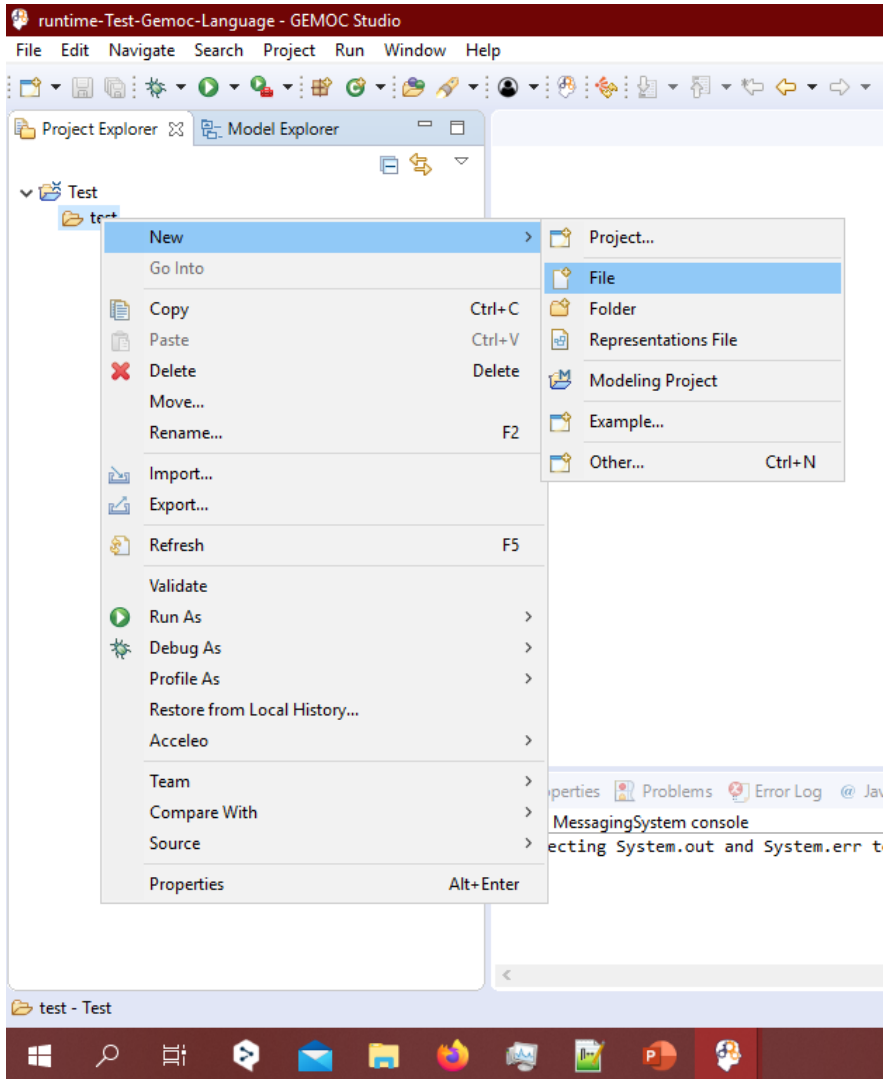
TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ



TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ



TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ

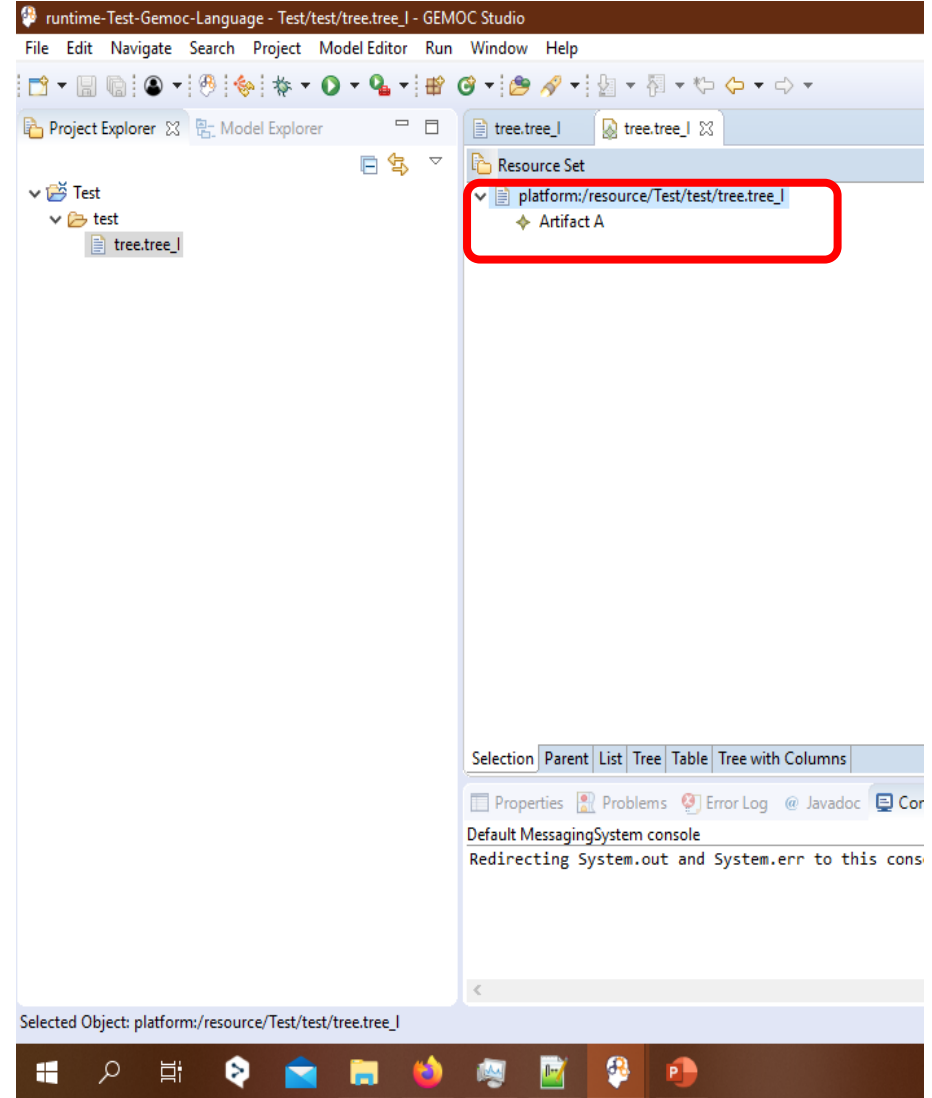
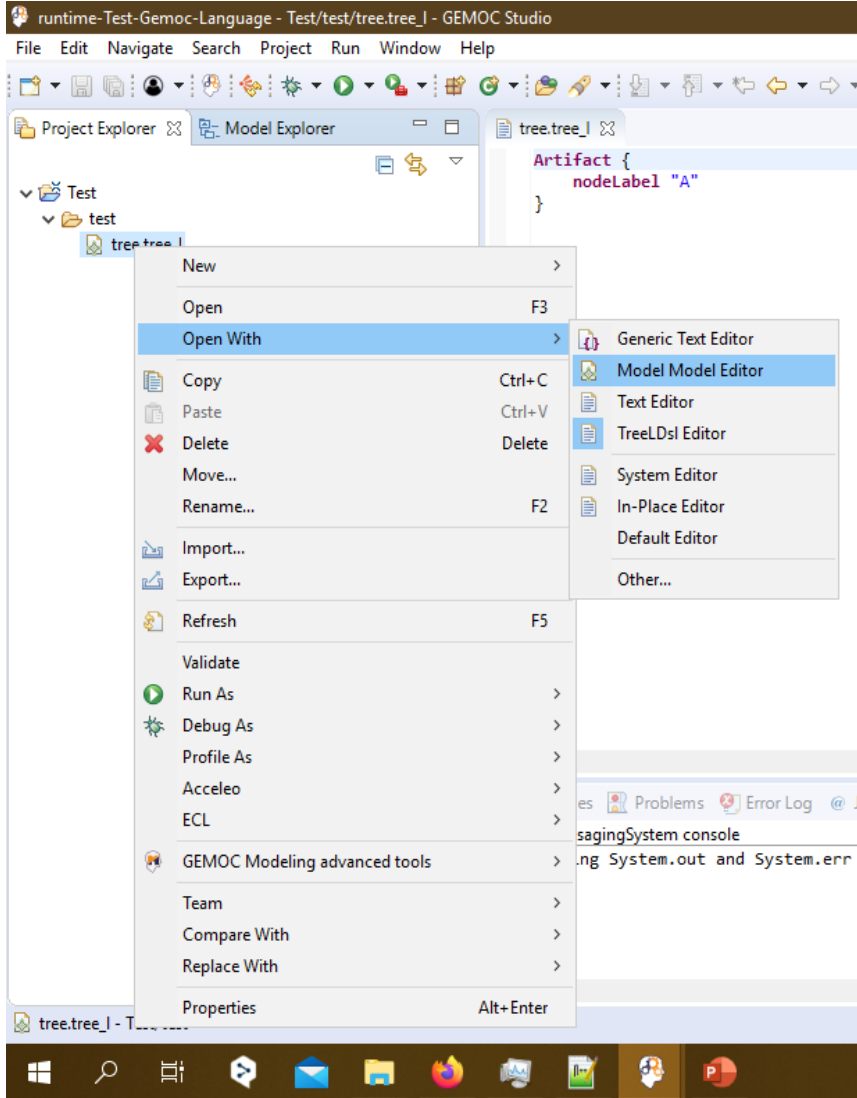
The screenshot displays the GEMOC Studio IDE interface. The main editor window shows the following code for an artifact:

```
Artifact {  
  nodeLabel "A"  
  sons (  
    nodeLabel "B"  
  )  
}
```

The Project Explorer on the left shows a project structure with a folder named 'test' containing a file named 'tree.tree_1'. The Outline view on the right shows a single node labeled 'A'. The Console view at the bottom displays the message: "Default MessagingSystem console. Redirecting System.out and System.err to this console." The status bar at the bottom indicates the current mode is 'Writable' and 'Insert', with a cursor at line 7, column 1. The Windows taskbar at the very bottom shows the system tray with the date and time set to 14:05 on 14/04/2020.

TRAVAIL PRATIQUE

7- UTILISATION DU LANGAGE MODÉLISÉ



PETIT EXTRA

Retrouvez une présentation de l'exemple déroulé dans ce document, sous forme de tutoriel vidéo ici



<https://drive.google.com/open?id=1OdYBvDFHPG2q6J4Pyhkt-8CAR2vd9E>

PERSPECTIVES ET CONCLUSION

- Dérouler un use case plus élaboré. Celui-ci pourrait être intéressant
→ http://gemoc.org/gemoc-studio-old/publish/tutorial_markedgraph/html_single/GuideTutorialMarkedGraph.html
- Apprendre à créer un syntaxe graphique avec Sirius
- Comprendre la notion de « sémantique opérationnelle » des langages, d'aspects, de coordination et d'exécution de ceux-ci
- GEMOC et GAG ????
- Etc.