**UMLx System Tutorial:**

1. **Source code structure**

data/: stores the UML diagrams collected from the selected projects, there might be more to be added in the future. The files with surfix .xml are exported from Enterprise Architect for the UML diagrams.

db\_backup/: the backup of current database, which is based on MongoDB.

documents/: the documents about the system configuration, tutorials of system structure, or functionality explanations.

Diagram\_profilers/: includes the basic operations to analyse the parsed UML diagrams. For example, UseCaseProcess.js includes the methods to identify transactions and categorize transactions from robustness diagrams, sequence diagrams, etc. DomainModelProcess.js includes the operations to analyse class diagrams.

Drivers/: includes the operations to deal with specific kinds of UML diagrams. The source code package is not currently used, since the operations are actually contained in Diagram\_profilers.

Evaluators/: this folder contains high level analysis for the extracted information from UML diagrams. For example, to use the number of transactions to calculate Use Case Point. Each of the JS files is an Evaluator, which is used as a plugin to register in UMLEvaluator.js.

Model\_platforms/: this folder contains the low level operations to parse a specific format of the exported files for UML models. For example, ea/XMI2.1Parser.js parses the information from the xml files exported from Enterprise Architect.

Node\_modules/: includes the installed nodejs plugins. We usually don’t touch the folder.

Public/: stores the uploaded data (the exported UML files) from the web portal, also the outputs from different analysis.

Rscript/: includes the R scripts for statistical analysis for the extracted information from UML models.

Temp/: store temporary files.

Views/: include templates for web pages, written with the Jade plugin of Nodejs.

UMLEstimator.js/: used to calibrate effort estimation models.

UMLEvaluator.js/: the abstraction for the evaluators. It currently registers the evaluators including: COCOMOCalculator.js, UMLModelEvaluator.js, and UseCaseCalculator.js.

UMLFileManager.js/: the UML model file manager module, which manages uploaded model files.

UMLModelInfoManagerMongoDB.js: includes the database management functions, for example, querying model or diagram information.

UMLxServices.js/: includes the Restful APIs to request the analytical data, including the Url access entries for the backend functionality.

1. **Programming Language:**

Nodejs, Javascript, HTML, CSS, R scripts

1. **Dependencies:**

NodeJs (with modules: multer, admzip, express,path,jade)

MongoDB

1. **Platform:**

Windows or Mac OS or Linux

1. **Setting up:**
2. install mongoDB. (https://docs.mongodb.com/manual/installation/)
3. restore the backup data base, using: mongorestore ./db\_backup
4. Install nodejs. (https://nodejs.org/en/download/)
5. Install nodejs packages using commands in terminal:
   1. npm install express
   2. npm install multer
   3. npm install adm-zip
   4. npm install path
   5. npm install jade
   6. npm install mongodb
   7. npm install eval
6. Start mongodb:
   1. Navigate to the database folder of MongoDB. The default folder for MongoDB installation is : ~/Documents/db.
   2. Using command to start mongoDB : mongod –dbpath .
7. Start web service:
   1. Navigate to your project root folder.
   2. Using command to start UMLxServices: node ./UMLxServices.js
8. Access: (<http://127.0.0.1:8081/>) to load the home page.

Notes: sometimes you may mess up the database for some reason. You can delete the mongodb and restore the backup database. This tutorial is just the skeleton of the process of setting up the project. There might be some variations for different operation systems and environments, or some of the instructions may not be very accurate. In those cases, you may need to do some researches for those steps.