Collaborative Software Development: from Goals to Coding

Pedro Monteiro

Information Systems and Computer Engineering
Instituto Superior Técnico
Supervisor: Prof. António Rito Silva

November 2019

Motivation

- Developers work on, and switch between different tasks throughout the day.
- There are many tools that support developers with the creation of tasks.
- The plugin Mylyn goes a step further and keeps track of each task's context.
- We want to take the next step and introduce the concept of activity context.

Research Question

In the frame of a software architecture, can the association of working context with a development activity foster the compliance with predefined development practices, and improve the developer's productivity?

- Activity: group of tasks and their artifacts
- Compliance: sequence of tasks in an Activity
- Performance: steps in the implementation /modification of an Activity

Related Work

References

Task Context and Patterns

- Mik Kersten and Gail C Murphy. Using task context to improve programmer productivity, 2006
- Benedikt Schmidt and Wolfgang Reinhardt. Task patterns to support task-centric social software engineering, 2009.
- Benedikt Schmidt and Uwe V Riss. Task patterns as means to experience sharing, 2009.
- Mik Kersten and Gail C Murphy. Task context for knowledge Workers, 2012.
- Mik Kersten and Gail C Murphy. Reducing friction for knowledge workers with task context, 2015.
- Murphy, Gail C and Kersten, Mik and Elves, Robert and Bryan, Nicole. Enabling Productive Software Development by Improving Information Flow, 2019

Work breakdown

- Thomas D LaToza, W Ben Towne, Christian M Adriano, and Andr Van Der Hoek. Microtask programming, 2014
- C Albert Thompson. Towards generation of software developmentTasks, 2015.
- C Albert Thompson, Gail C Murphy, Marc Palyart, and Marko Gaparic. How software developers use work breakdown relationships in issue repositories, 2016.
- Walid Maalej, Mathias Ellmann, and Romain Robbes. Using contexts similarity to predict relationships between tasks, 2017.
- Christoph Mayr-Dorn and Alexander Egyed. Does the propagation of artifact changes across tasks reflect work dependencies? 2018.

Task Context vs Task Patterns vs Activity Context

Task Context (Mylyn)

- Degree-of-interest algorithm
- Task list with task activation that focuses workspace
- Strict focus on reorganizing the current information
- Our focus is on the whole activity and facilitating the flow between tasks, while maintaining compliance with the architecture and development practice.

Task Patterns

- Task execution creates a pattern
- Focus on collaboration
- Re-usable knowledge structure
- Guides by reuse
- We guide by definition, with the structuring of an activity driven by the software architecture and development practice

Problems

- No task structure
- No support for new tasks
- Random context changes
- No task suggestion
- Limited data-awareness

Objectives

- Data Model (data awareness)
- Metadata Model (structure for creation of context)
- Activity Context (implementation/modification)
- Compliance with the architecture
- Compliance with the development practice
- Productivity increase

Solution

Development Practice

Represented in a Metadata template

- Task Types, Dependency Types and Artifact Types.

Software Architecture

Represented in a Data template

- Artifacts and Dependencies

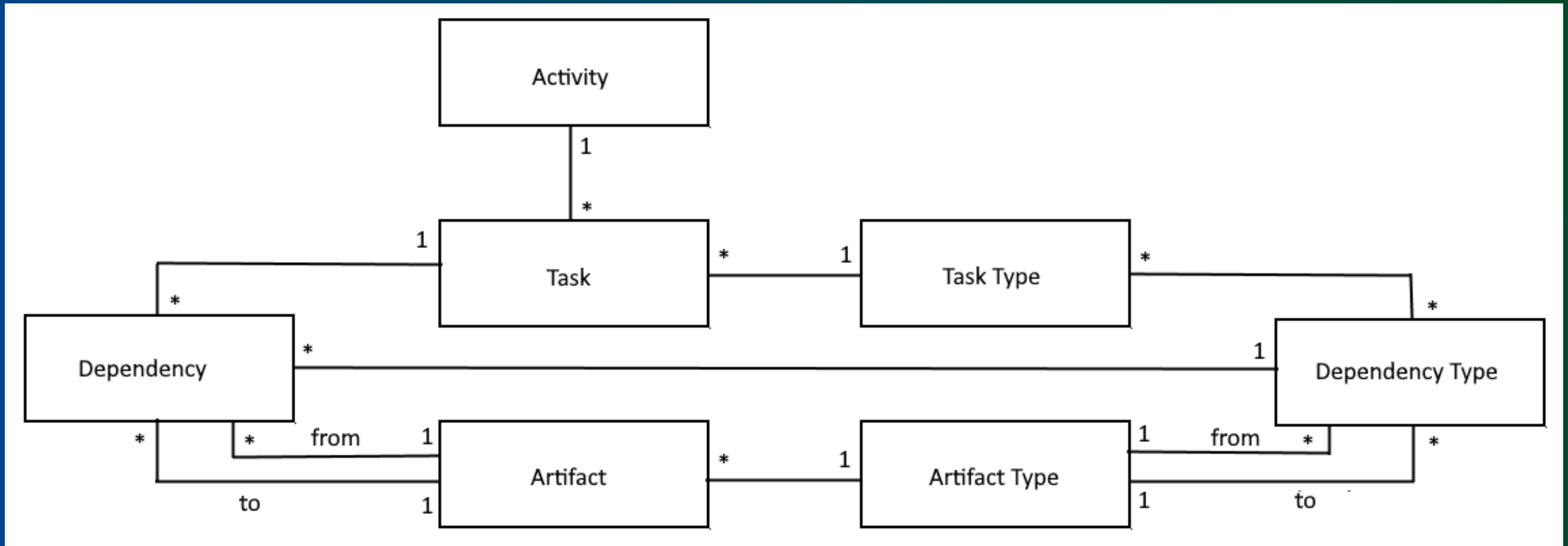


Figure 1

Solution

Activity

- Structured by the software architecture and the development practice.
- Has a context: the group of tasks and their artifacts.
- Automatic features, such as task suggestion
- Self contained

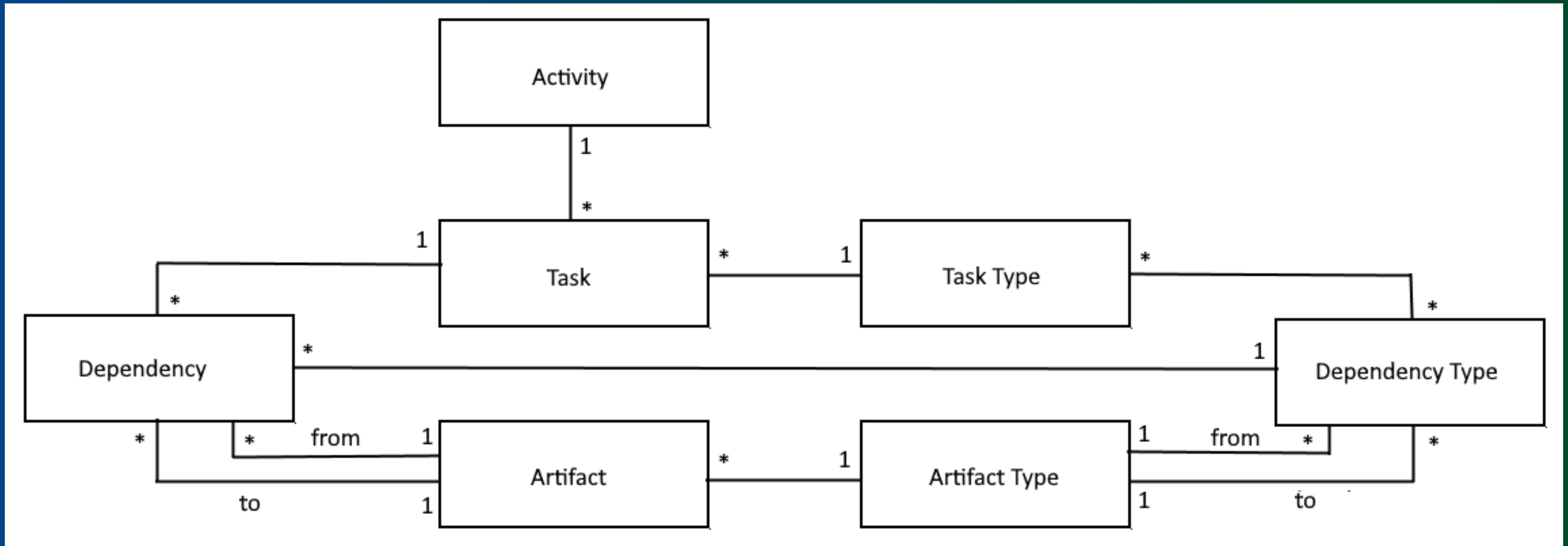


Figure 1

Solution

Task

- Transformation of artifacts
- Has a task type
- Has a context: the artifacts
- Has a state

Task Type

- Group of dependency types
- Has a main dependency type

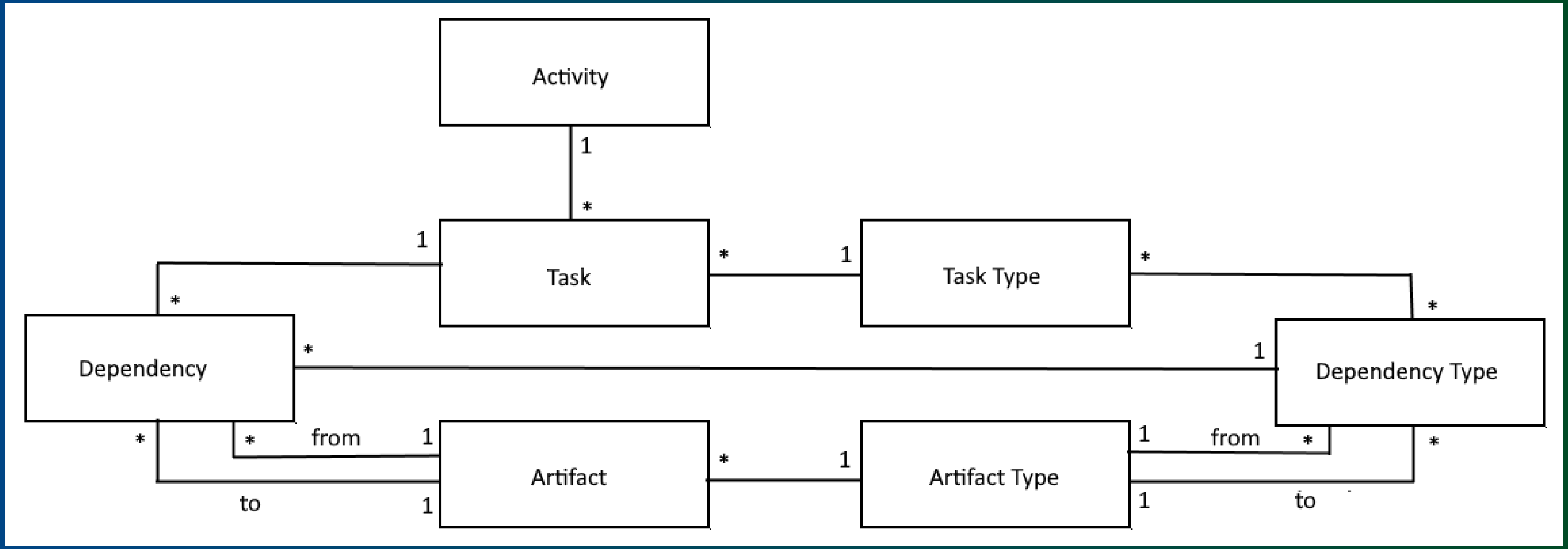


Figure 1

Solution

Dependency

- Relation between two artifacts.
- Has a dependency type.

Dependency Type

- Relation between two artifact types, where the first is needed for the second.

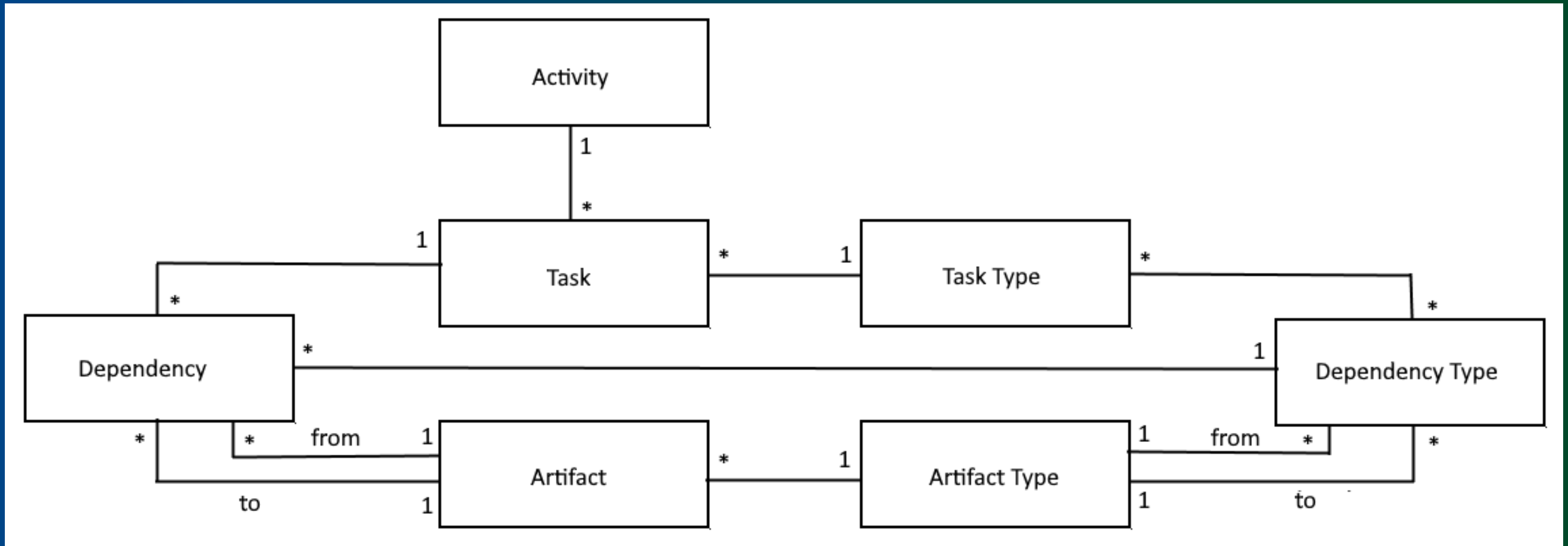


Figure 1

Solution

Artifact

- File in the workspace
- Has an artifact type
- Has a binary state
- Contained in a package

Artifact Type

- Represents artifacts

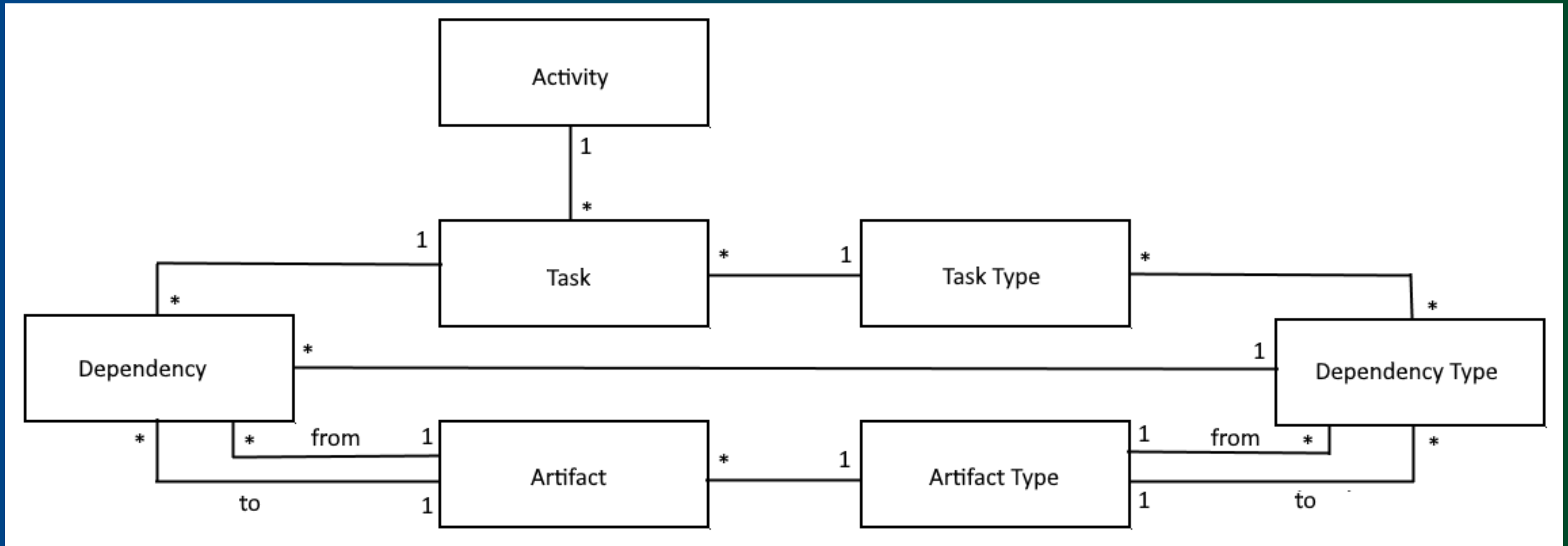


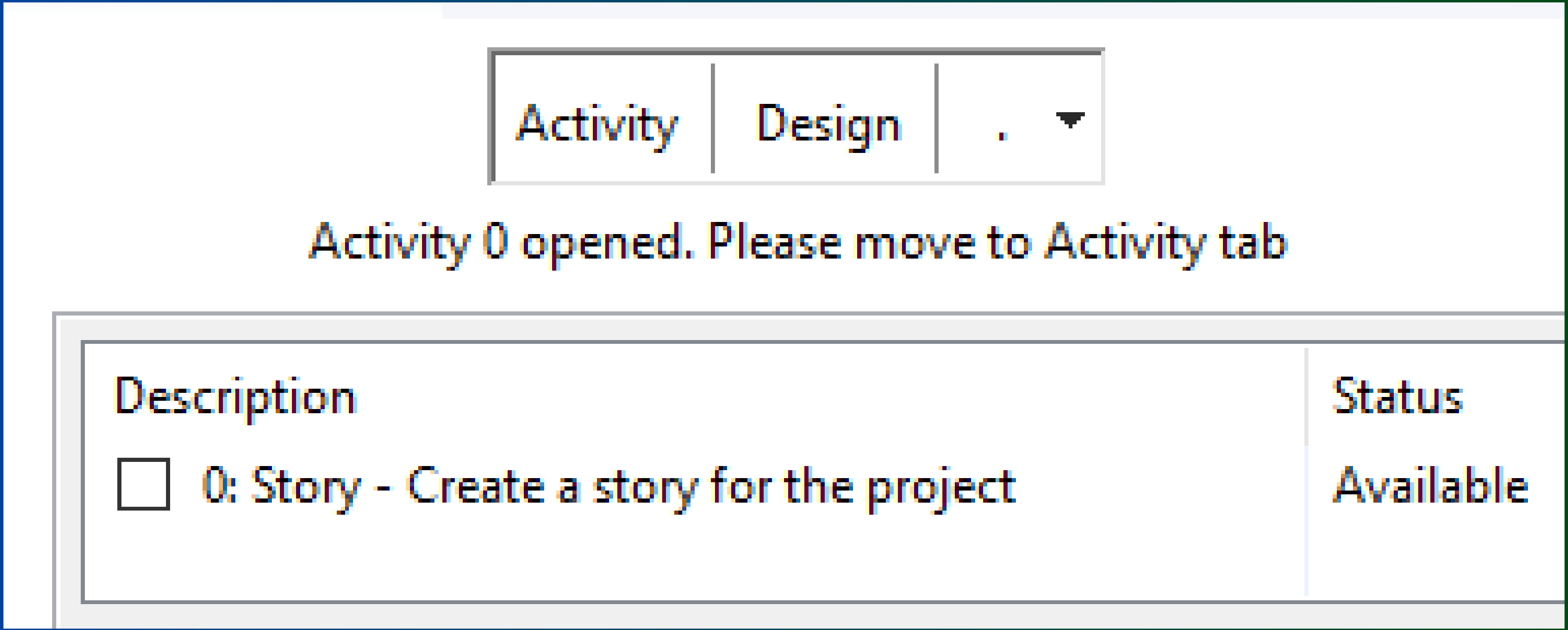
Figure 1

Archlyn

Workflow

- Workflow A - The developer works on an already supported project.
    - Implementing a new activity
    - Modifying an existing activity
- Workflow B – Create the metadata template
- Workflow C – Create the data template

Workflow A – Activity View

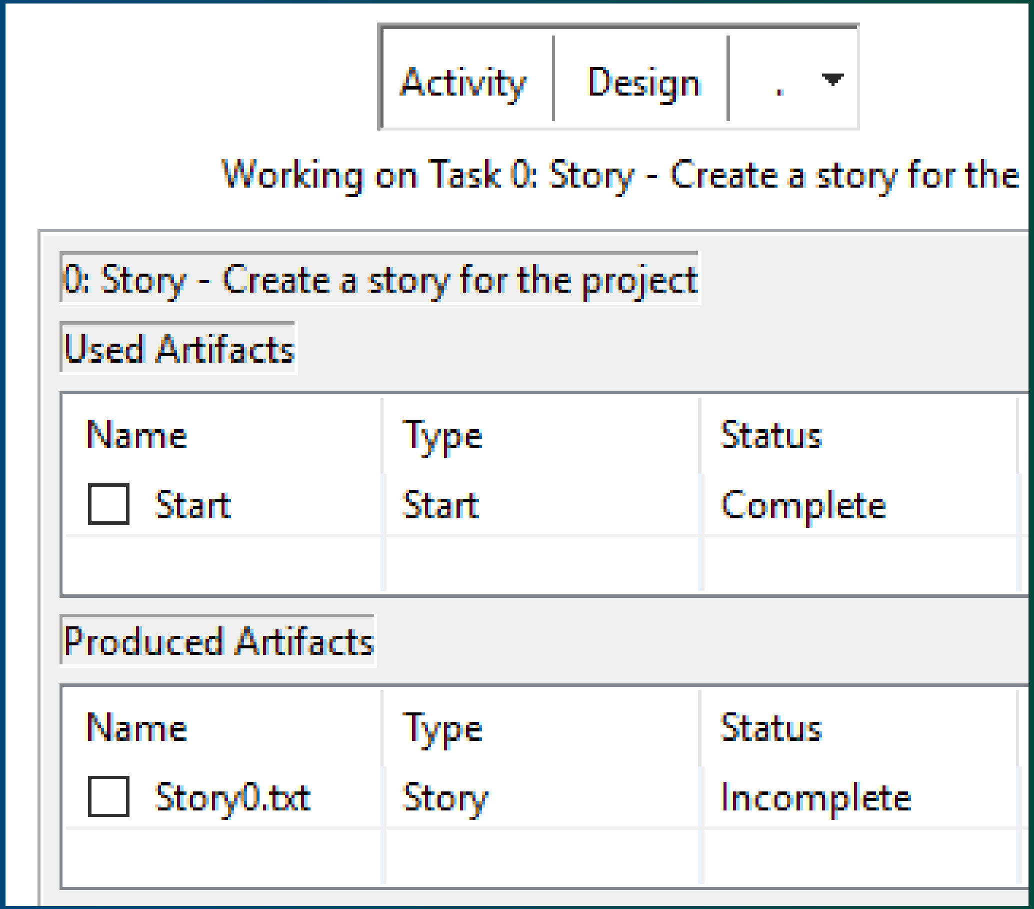


Activity 0 opened. Please move to Activity tab

Description	Status
□ 0: Story - Create a story for the project	Available

Figure 2

Workflow A – Task View



Working on Task 0: Story - Create a story for the

0: Story - Create a story for the project

Used Artifacts

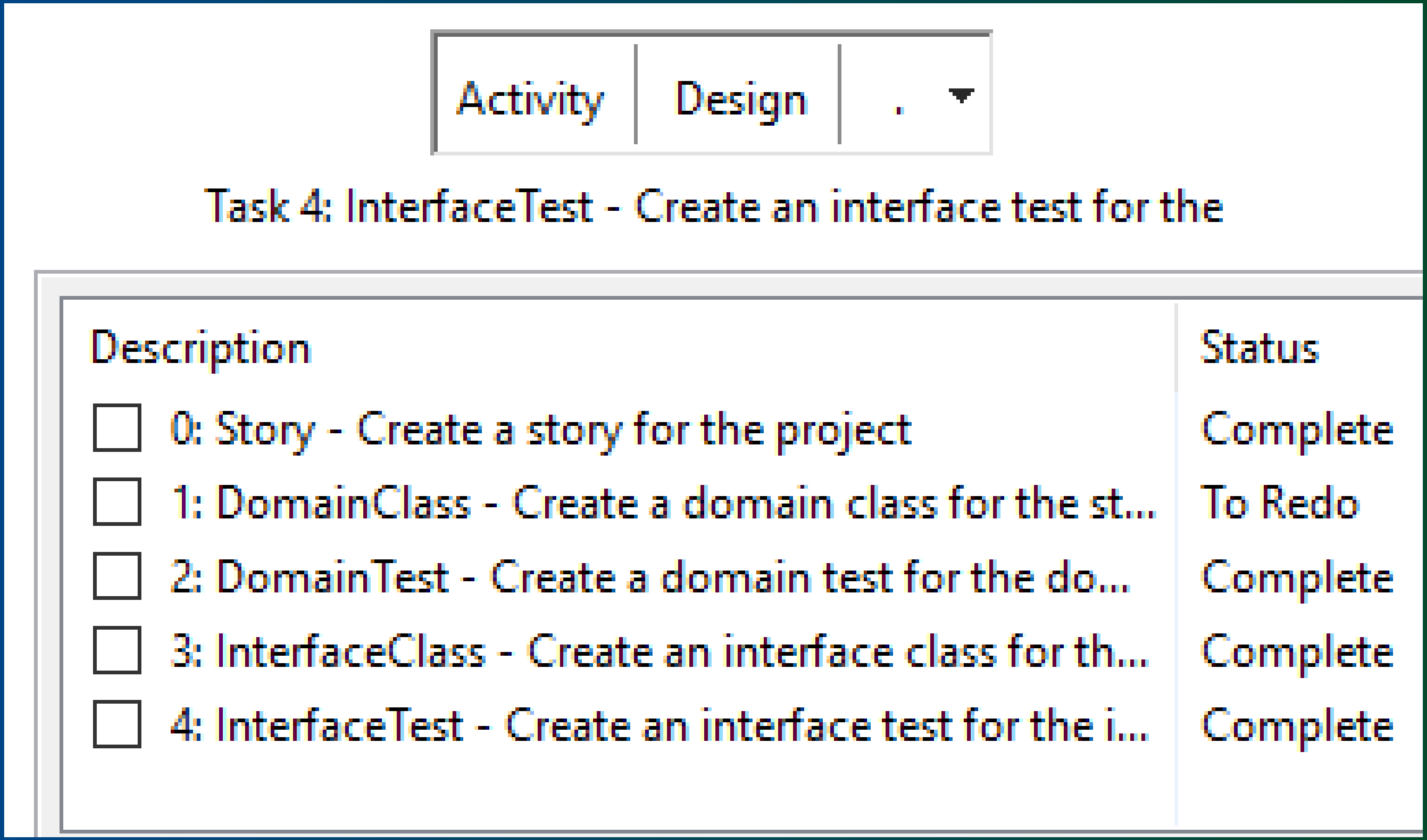
Name	Type	Status
□ Start	Start	Complete

Produced Artifacts

Name	Type	Status
□ Story0.txt	Story	Incomplete

Figure 3

Workflow A - Reactivation

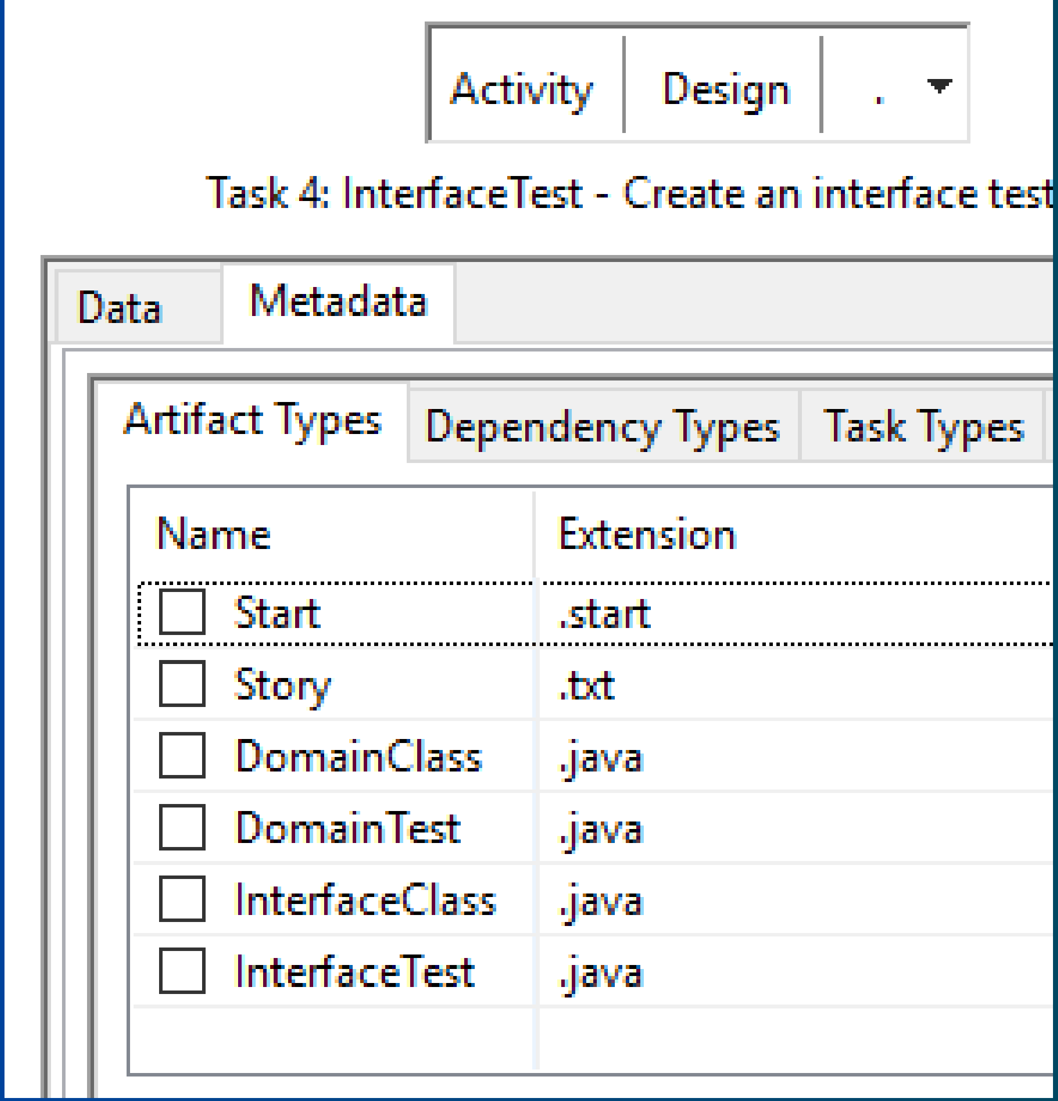


Task 4: InterfaceTest - Create an interface test for the

Description	Status
□ 0: Story - Create a story for the project	Complete
□ 1: DomainClass - Create a domain class for the st...	To Redo
□ 2: DomainTest - Create a domain test for the do...	Complete
□ 3: InterfaceClass - Create an interface class for th...	Complete
□ 4: InterfaceTest - Create an interface test for the i...	Complete

Figure 5

Workflow B

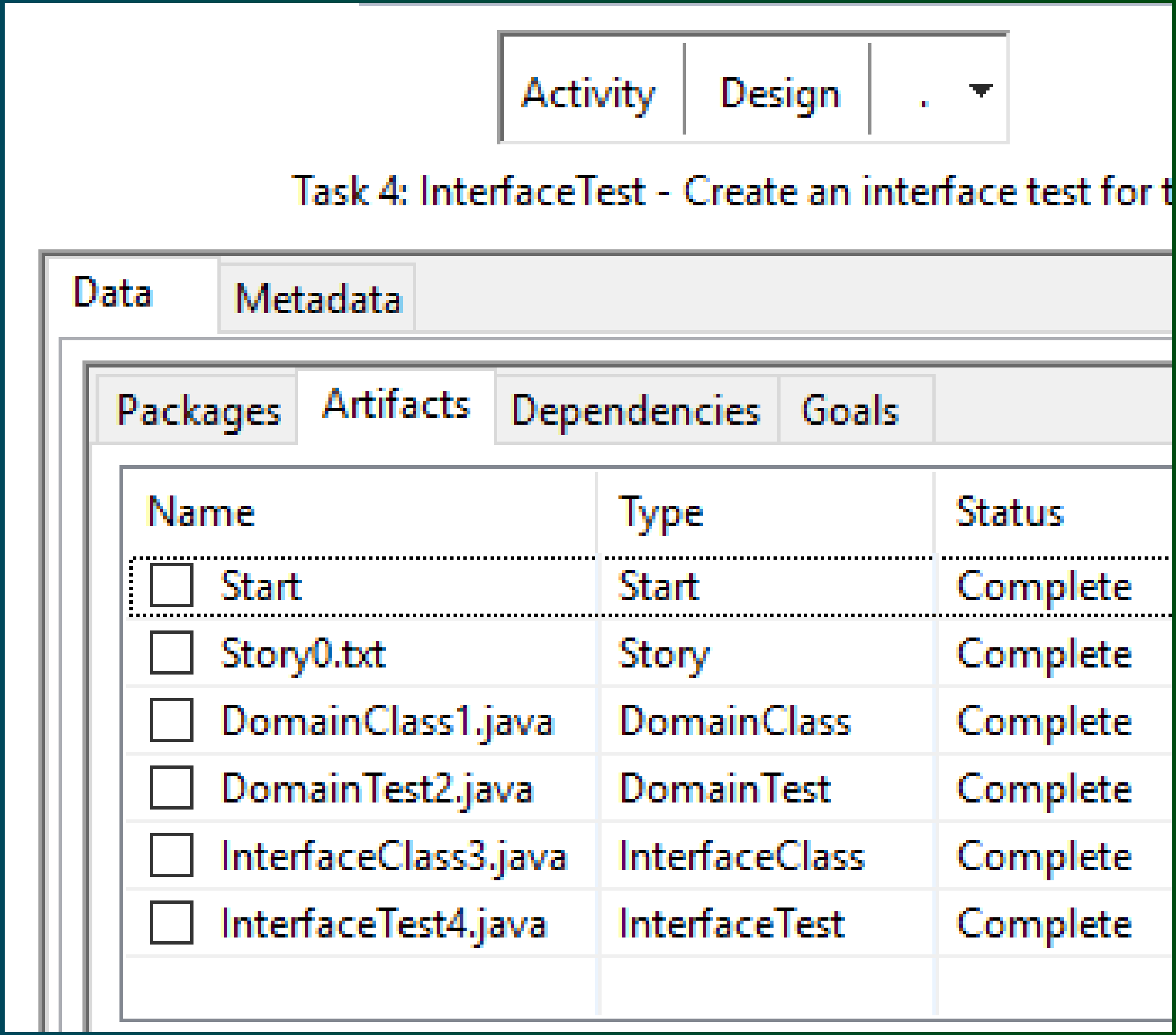


Task 4: InterfaceTest - Create an interface test

Name	Extension
□ Start	.start
□ Story	.txt
□ DomainClass	.java
□ DomainTest	.java
□ InterfaceClass	.java
□ InterfaceTest	.java

Figure 6

Workflow C



Task 4: InterfaceTest - Create an interface test for t

Name	Type	Status
□ Start	Start	Complete
□ Story0.txt	Story	Complete
□ DomainClass1.java	DomainClass	Complete
□ DomainTest2.java	DomainTest	Complete
□ InterfaceClass3.java	InterfaceClass	Complete
□ InterfaceTest4.java	InterfaceTest	Complete

Figure 7

Evaluation

Activity 1: Learning

- Task 1 - Creation of a Story
- Task 2 - Creation of a DomainClass
- Task 3 - Creation of a DomainTest
- Task 4 - Creation of an InterfaceClass
- Task 5 - Creation of an InterfaceTest

Evaluation

Activity 1 – Task steps

- Activate the available task
- Move the created artifact to the source folder
- The code is provided to the user
- Rename the artifact according to the story
- Complete the active task

Evaluation

Activity 2: Test implementation workflow

- Task 1 - Creation of a Story (New Story)
- Task 2 - Creation of a DomainClass
- ~~Task 3 - Creation of a DomainTest~~
- Task 4 - Creation of an InterfaceClass
- ~~Task 5 - Creation of an InterfaceTest~~

Evaluation

Activity 3: Test reactivation workflow

- Task 1 – Update Story (New Story)
- Task 2 - Update DomainClass
- Task 3 – Update DomainTest
- Task 4 – Update InterfaceClass
- Task 5 – Update InterfaceTest

Evaluation

Hypothesis

"By working in an activity context we can achieve a very high percentage of compliance with the software architecture, a high percentage of compliance with the development practice, and a low variation in user performance resulting from the fostered process.”

Compliance

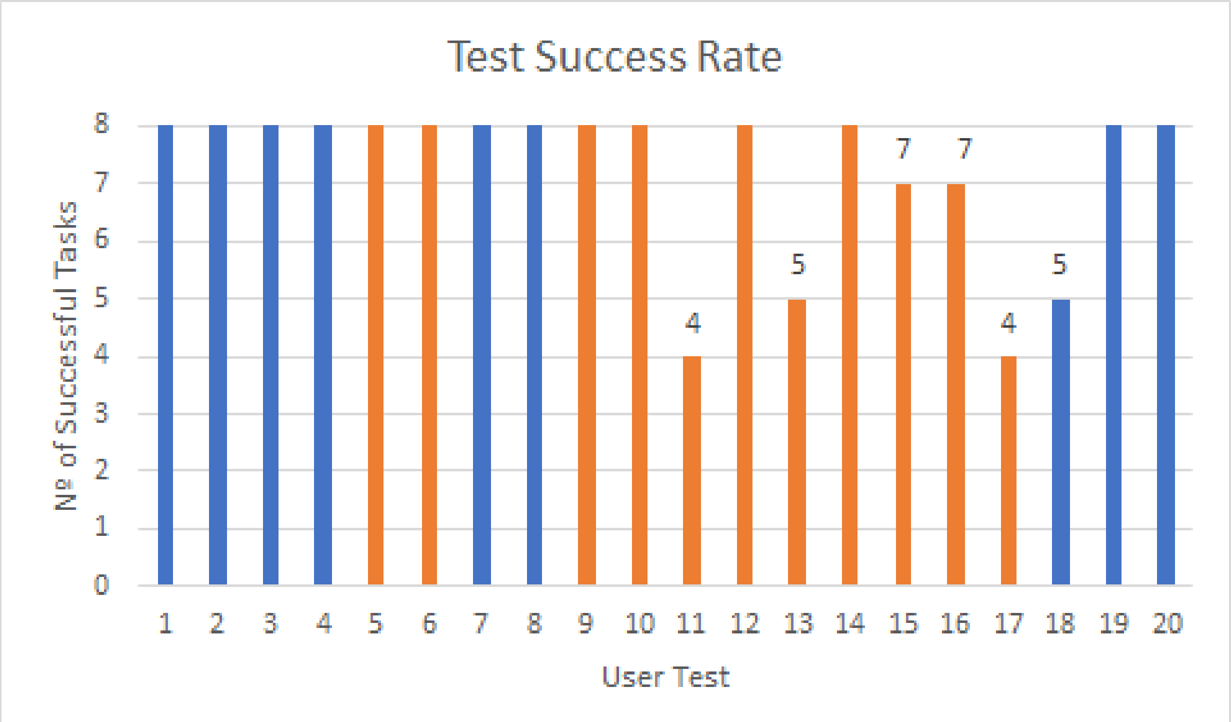


Figure 8

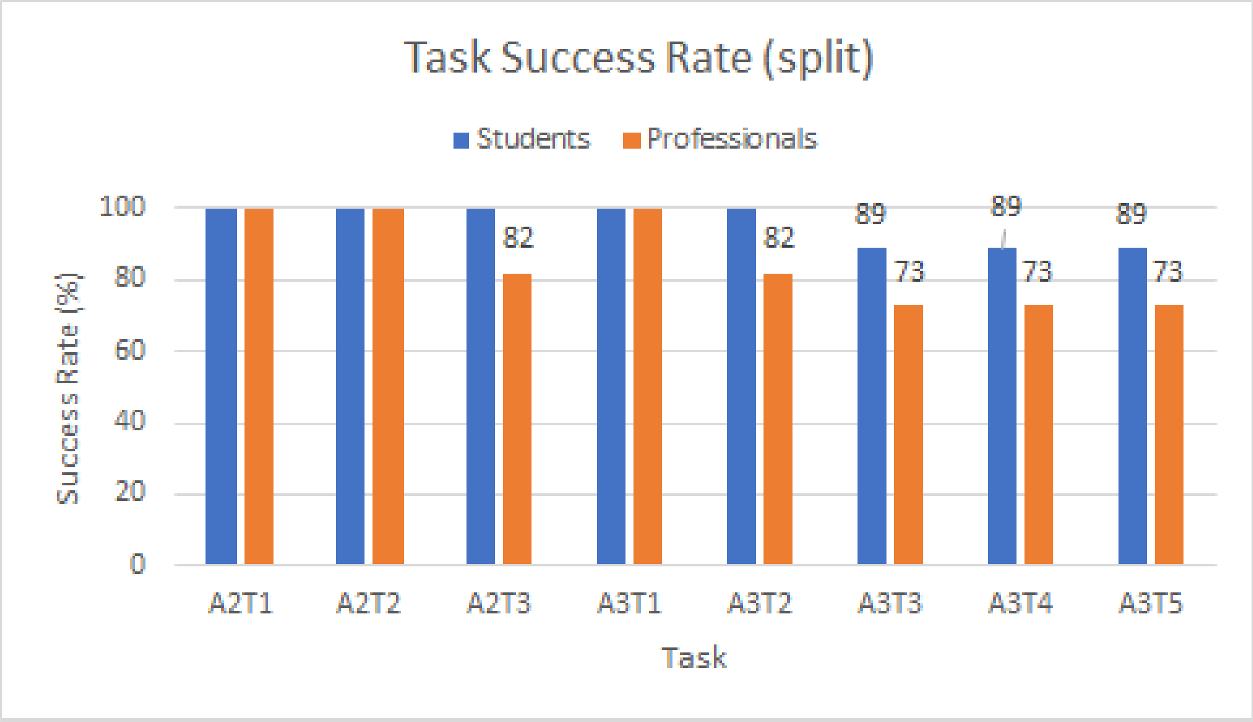


Figure 9

Performance

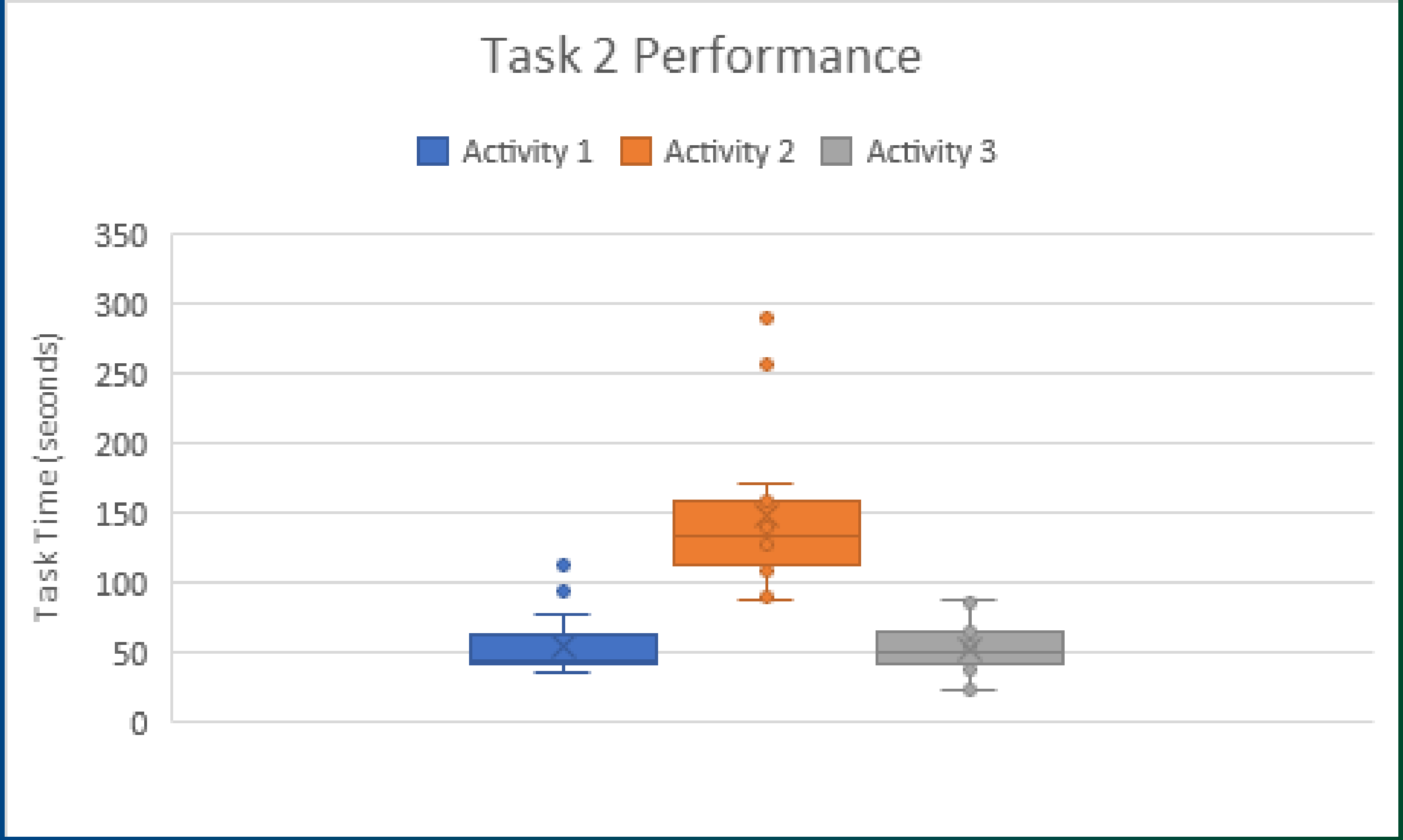


Figure 10

Conclusion

- 100% compliance with the software architecture
- 70% compliance with the development practice
- Low deviation in user performance
- Versatile approach
- Ideal for projects that value fostering compliance

Future Work

- Does the Plugin restrict freedom of development?
- Delve into code level: Customize an artifact's code.
- Context sharing and automatic task attribution
- Gamify the activity workflow

THANK YOU!

Pedro Monteiro