REENGINEERING CI/CD PIPELINES

A TWO-LEVEL MODEL-DRIVEN ENGINEERING APPROACH

André Flores

Supervisor: Jácome Cunha
Co-supervisor: Hugo Gião

03/07/2024

WHAT ARE CI/CD PIPELINES?

WHAT ARE CI/CD PIPELINES?

Continuous Integration, Delivery, and Deployment

WHAT ARE CI/CD PIPELINES?

Continuous Integration, Delivery, and Deployment

- Means changes to a program's code are integrated into the system and deployed to production with little delay[1,2].

WHAT ARE CI/CD PIPELINES?

Continuous Integration, Delivery, and Deployment

- Means changes to a program's code are integrated into the system and deployed to production with little delay[1,2].
- Practicing CI/CD implies frequent building, testing, and deployment of the system.

WHAT ARE CI/CD PIPELINES?

Continuous Integration, Delivery, and Deployment

- Means changes to a program's code are integrated into the system and deployed to production with little delay[1,2].
- Practicing CI/CD implies frequent building, testing, and deployment of the system.
- These activities are organized into CI/CD pipelines.

WHAT ARE CI/CD PIPELINES?

Continuous Integration, Delivery, and Deployment

- Means changes to a program's code are integrated into the system and deployed to production with little delay[1,2].
- Practicing CI/CD implies frequent building, testing, and deployment of the system.
- These activities are organized into CI/CD pipelines.
- Pipelines are implemented in one or more CI/CD platforms like GitHub Actions, CircleCI, and Jenkins.

WHAT ARE CI/CD PIPELINES?

Continuous Integration, Delivery, and Deployment

- Means changes to a program's code are integrated into the system and deployed to production with little delay[1,2].
- Practicing CI/CD implies frequent building, testing, and deployment of the system.
- These activities are organized into CI/CD pipelines.
- Pipelines are implemented in one or more CI/CD platforms like GitHub Actions, CircleCI, and Jenkins.
- Projects migrate CI/CD pipelines.

PROBLEMS WHEN MIGRATING CD/CD TECHNOLOGIES[4]

PROBLEMS WHEN MIGRATING CD/CD TECHNOLOGIES[4]

FUNDAMENTAL DIFFERENCES BETWEEN TECHNOLOGIES

PROBLEMS WHEN MIGRATING CD/CD TECHNOLOGIES[4]

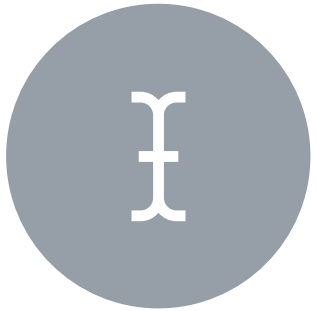
FUNDAMENTAL DIFFERENCES BETWEEN TECHNOLOGIES

CONFIGURING CI/CD IS TRIAL-AND-ERROR BY NATURE

PROBLEMS WHEN MIGRATING CD/CD TECHNOLOGIES[4]

FUNDAMENTAL DIFFERENCES BETWEEN TECHNOLOGIES

CONFIGURING CI/CD IS TRIAL-AND-ERROR BY NATURE

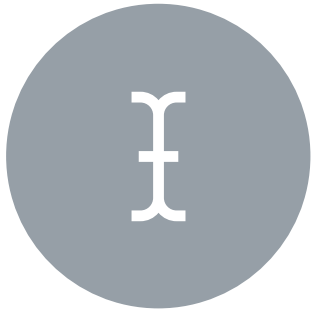


DIFFICULTIES WITH SYNTAX

PROBLEMS WHEN MIGRATING CD/CD TECHNOLOGIES[4]

FUNDAMENTAL DIFFERENCES BETWEEN TECHNOLOGIES

CONFIGURING CI/CD IS TRIAL-AND-ERROR BY NATURE



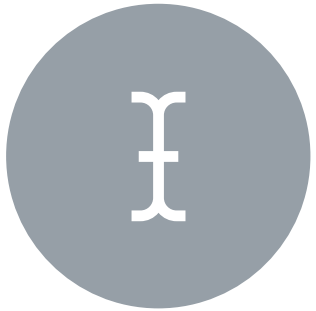
DIFFICULTIES WITH SYNTAX

LACK OF SUPPORT FROM PROVIDERS

PROBLEMS WHEN MIGRATING CD/CD TECHNOLOGIES[4]

FUNDAMENTAL DIFFERENCES BETWEEN TECHNOLOGIES

CONFIGURING CI/CD IS TRIAL-AND-ERROR BY NATURE



DIFFICULTIES WITH SYNTAX

LACK OF SUPPORT FROM PROVIDERS

Migration is long and arduous[5]

HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

A CI/CD Pipeline Transpiler

HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

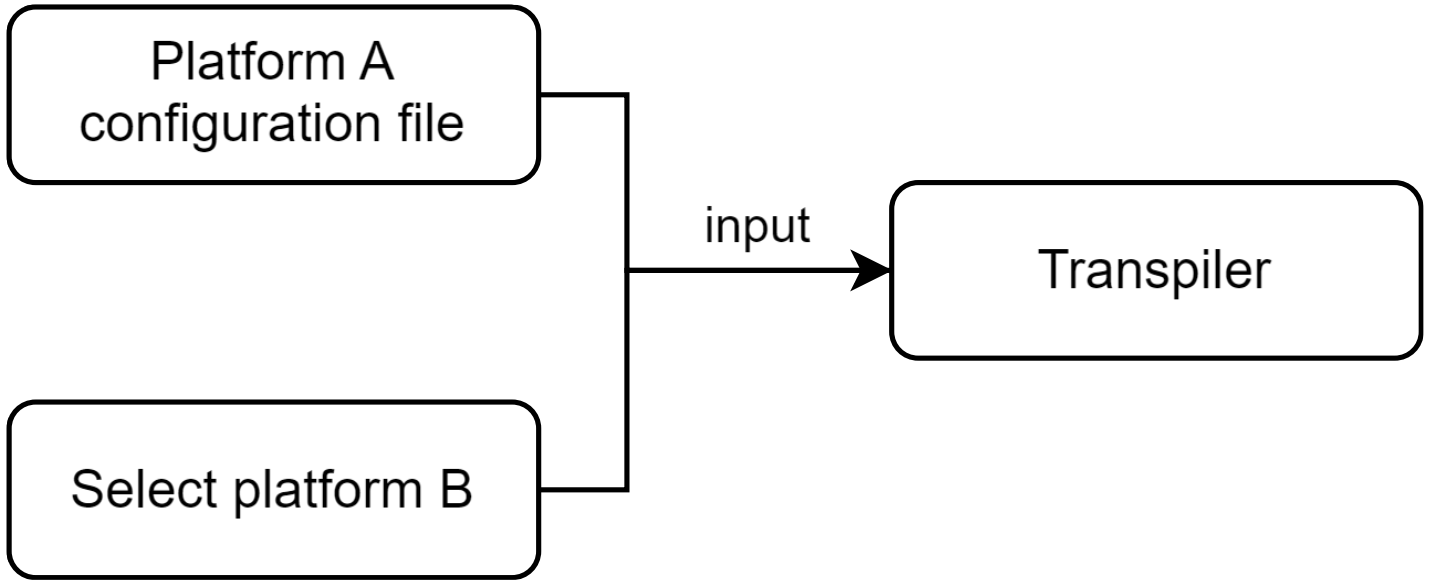
A CI/CD Pipeline Transpiler

Platform A configuration file

Select platform B

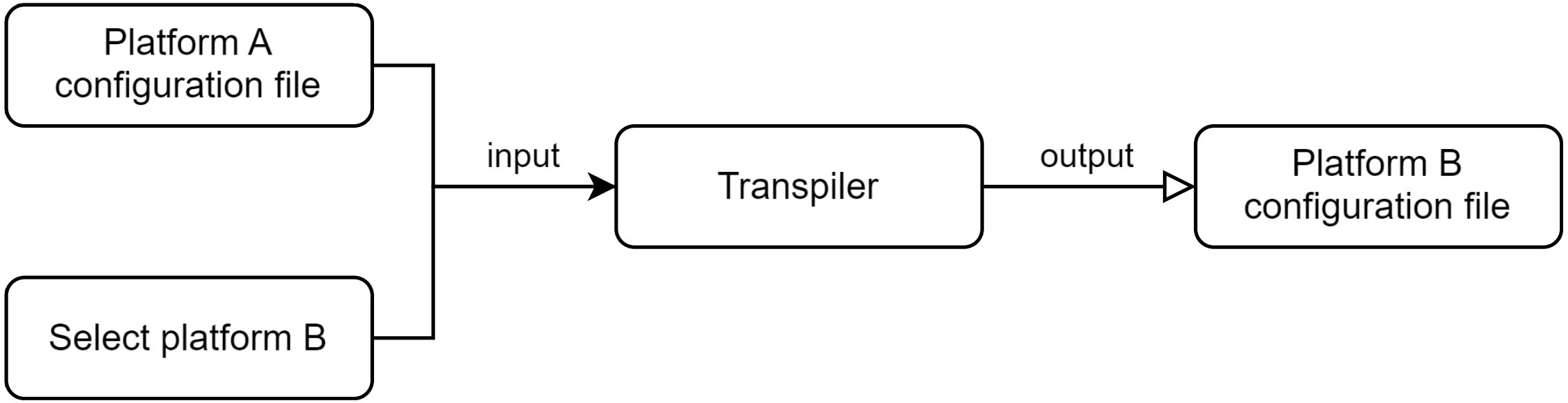
HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

A CI/CD Pipeline Transpiler



HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

A CI/CD Pipeline Transpiler



HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

A CI/CD Pipeline Transpiler

- Simplifies and speeds up migration.

HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

A CI/CD Pipeline Transpiler

- Simplifies and speeds up migration.
- Helps developers keep up productivity.

HOW DO WE HELP DEVELOPERS MIGRATE CI/CD?

A CI/CD Pipeline Transpiler

- Simplifies and speeds up migration.
- Helps developers keep up productivity.
- Reduces lock-in to CI/CD platforms.

RESEARCH QUESTIONS

RESEARCH QUESTIONS

RQ1: What are the main core concepts shared by and unique to the different CI/CD platforms?

RESEARCH QUESTIONS

RQ1: What are the main core concepts shared by and unique to the different CI/CD platforms?

RQ2: Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

RESEARCH QUESTIONS

RQ1: What are the main core concepts shared by and unique to the different CI/CD platforms?

RQ2: Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

RQ3: Can CI/CD pipeline migration be fully automated?

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

Automated migration software

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

Automated migration software

- Supports migration from seven CI/CD platforms to GitHub Actions.

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

Automated migration software

- Supports migration from seven CI/CD platforms to GitHub Actions.
- Limitations vary with the platform being migrated from, they are related to functionality and unknown packages.

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

Automated migration software

- Supports migration from seven CI/CD platforms to GitHub Actions.
- Limitations vary with the platform being migrated from, they are related to functionality and unknown packages.
- Only supports migration to GitHub Actions.

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

Automated migration software

- Supports migration from seven CI/CD platforms to GitHub Actions.
- Limitations vary with the platform being migrated from, they are related to functionality and unknown packages.
- Only supports migration to GitHub Actions.
- Has a DSL to increase functionality.

INDUSTRY – GITHUB ACTIONS IMPORTER[7]

Automated migration software

- Supports migration from seven CI/CD platforms to GitHub Actions.
- Limitations vary with the platform being migrated from, they are related to functionality and unknown packages.
- Only supports migration to GitHub Actions.
- Has a DSL to increase functionality.
- Goal is an 80% conversion rate.

RESEARCH – MODELING AND CI/CD[8]

RESEARCH – MODELING AND CI/CD[8]

DOCUMENTATION AND GUIDANCE

9, 18, 19, 27

RESEARCH – MODELING AND CI/CD[8]

DOCUMENTATION AND GUIDANCE

9, 18, 19, 27

CLOUD/IoT DEPLOYMENT

10, 11, 12, 13, 14, 15, 16, 17, 21, 22, 23, 24, 25, 28, 29, 30 31, 32, 33

RESEARCH – MODELING AND CI/CD[8]

DOCUMENTATION AND GUIDANCE

9, 18, 19, 27

CLOUD/IoT DEPLOYMENT

10, 11, 12, 13, 14, 15, 16, 17, 21, 22, 23, 24, 25, 28, 29, 30 31, 32, 33

DATA SCIENCE PIPELINES

20

RESEARCH – MODELING AND CI/CD[8]

DOCUMENTATION AND GUIDANCE

9, 18, 19, 27

CLOUD/IoT DEPLOYMENT

10, 11, 12, 13, 14, 15, 16, 17, 21, 22, 23, 24, 25, 28, 29, 30 31, 32, 33

DATA SCIENCE PIPELINES

20

SPECIFIC CI/CD PLATFORMS

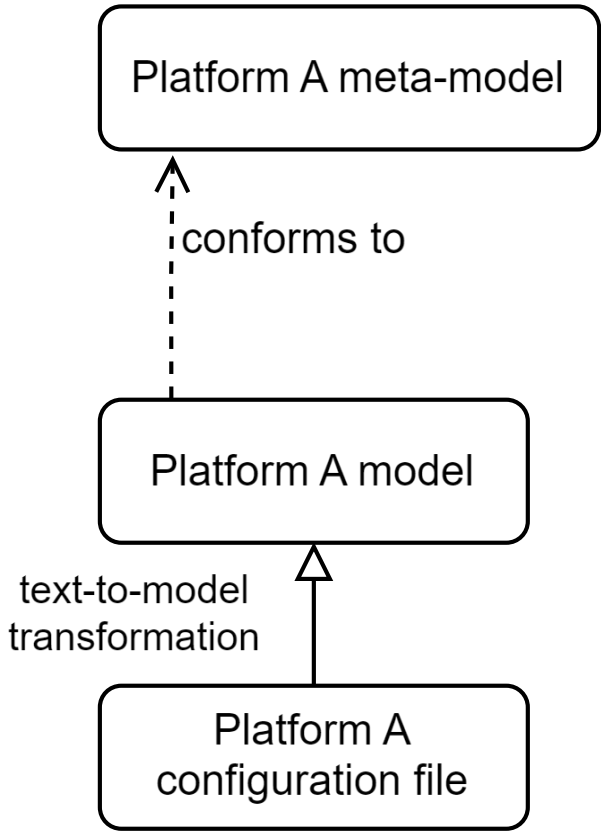
26

THE CI/CD PIPELINE REENGINEERING PROCESS

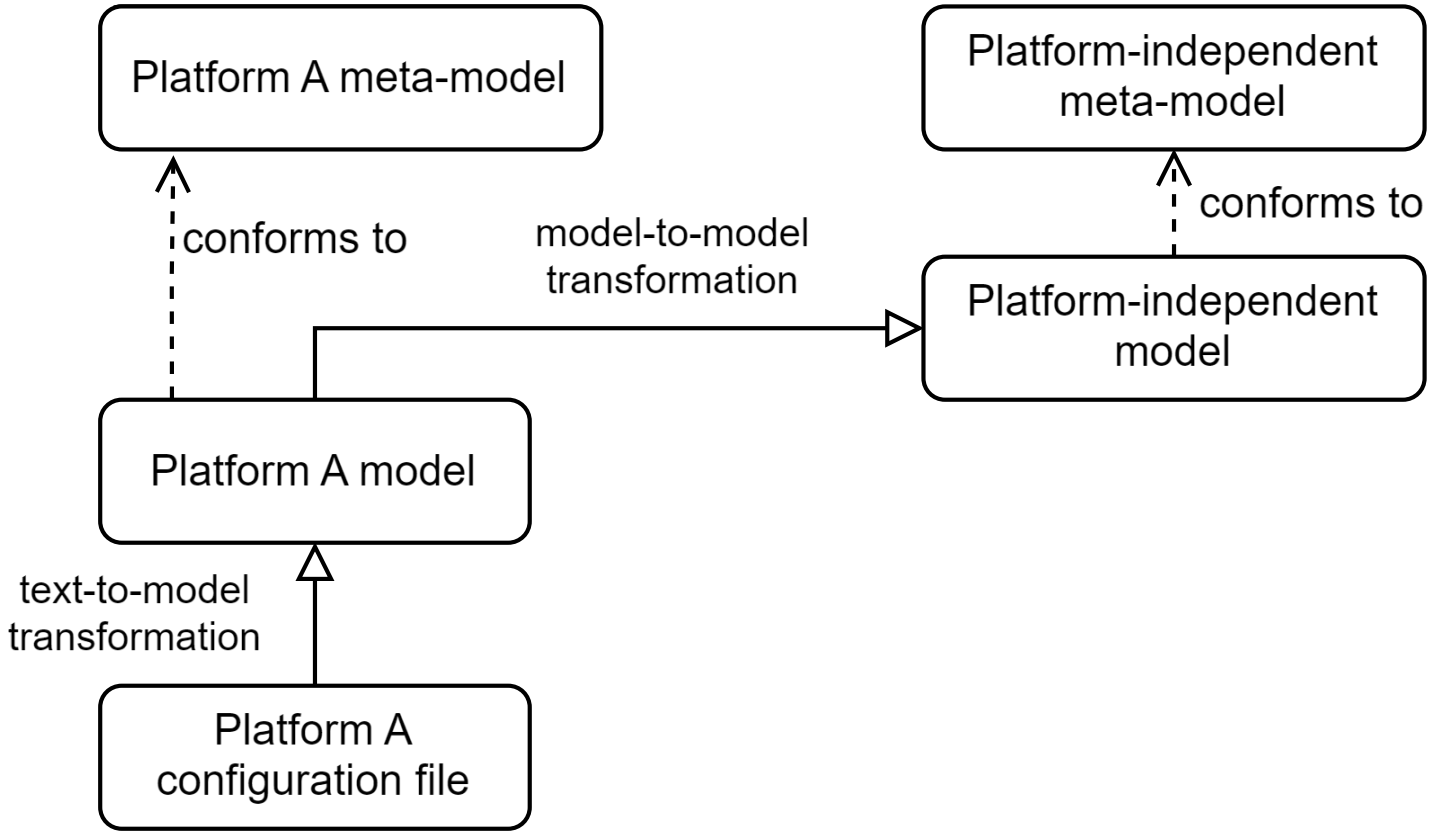
THE CI/CD PIPELINE REENGINEERING PROCESS

Platform A
configuration file

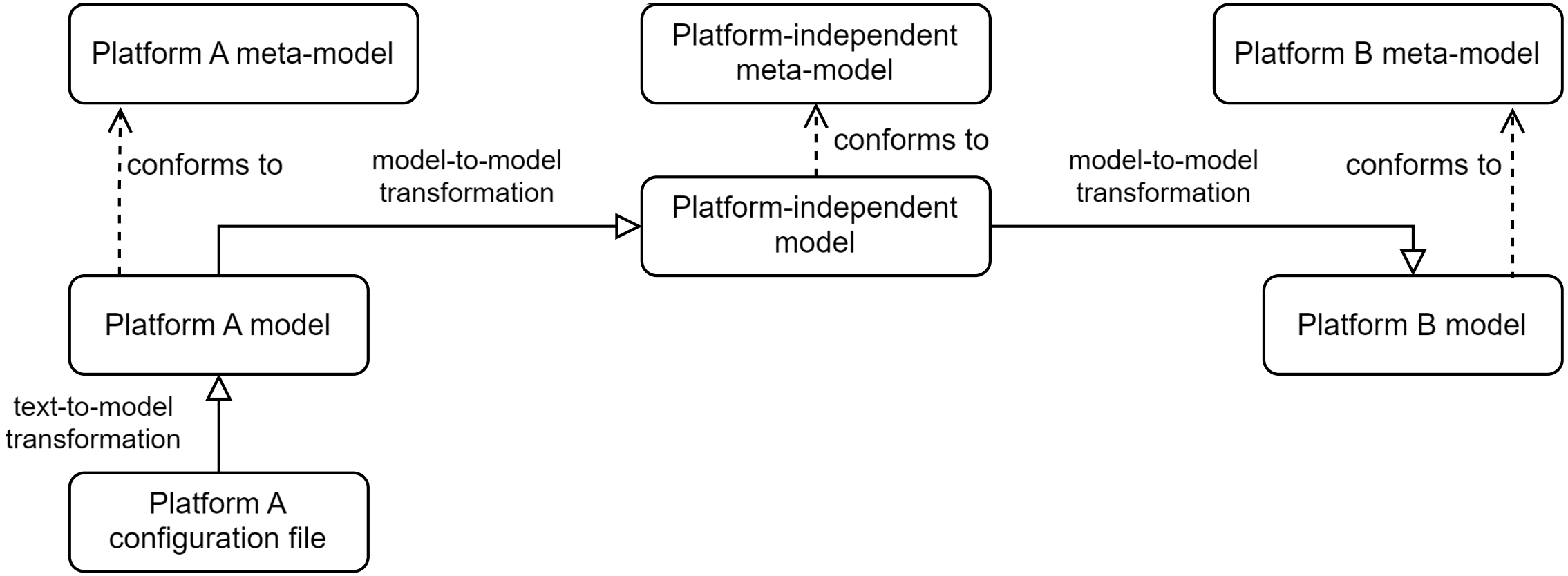
THE CI/CD PIPELINE REENGINEERING PROCESS



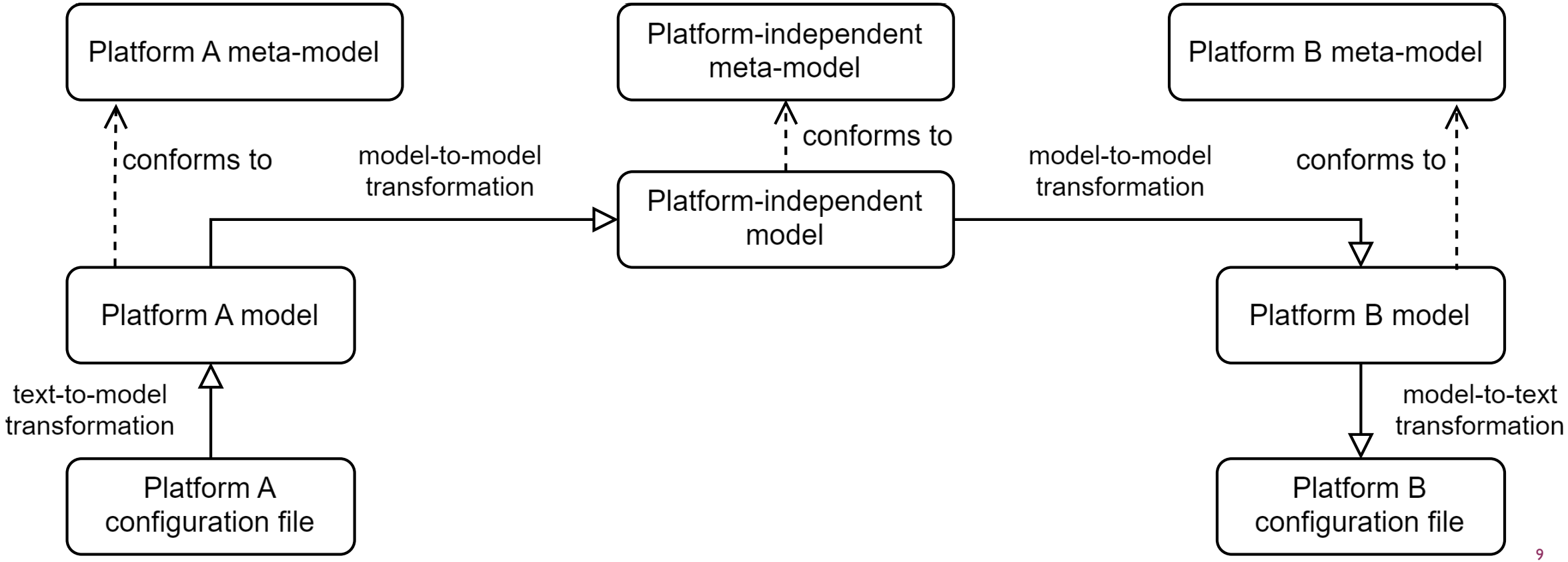
THE CI/CD PIPELINE REENGINEERING PROCESS



THE CI/CD PIPELINE REENGINEERING PROCESS



THE CI/CD PIPELINE REENGINEERING PROCESS



THE PLATFORM-SPECIFIC META-MODELS – ADDRESSING RQ1

THE PLATFORM-SPECIFIC META-MODELS – ADDRESSING RQ1

What are the main core concepts shared by and unique to the different CI/CD platforms?

THE PLATFORM-SPECIFIC META-MODELS – ADDRESSING RQ1

What are the main core concepts shared by and unique to the different CI/CD platforms?

- Created platform-specific meta-models for 3 popular CI/CD platforms – GitHub Actions, CircleCI, and Jenkins.

THE PLATFORM-SPECIFIC META-MODELS – ADDRESSING RQ1

What are the main core concepts shared by and unique to the different CI/CD platforms?

- Created platform-specific meta-models for 3 popular CI/CD platforms – GitHub Actions, CircleCI, and Jenkins.
- Close to the textual syntax of the platforms so there is no loss of concepts.

THE PLATFORM-SPECIFIC META-MODELS – ADDRESSING RQ1

What are the main core concepts shared by and unique to the different CI/CD platforms?

- Created platform-specific meta-models for 3 popular CI/CD platforms – GitHub Actions, CircleCI, and Jenkins.
- Close to the textual syntax of the platforms so there is no loss of concepts.
- The platforms revealed to have several core, common concepts.

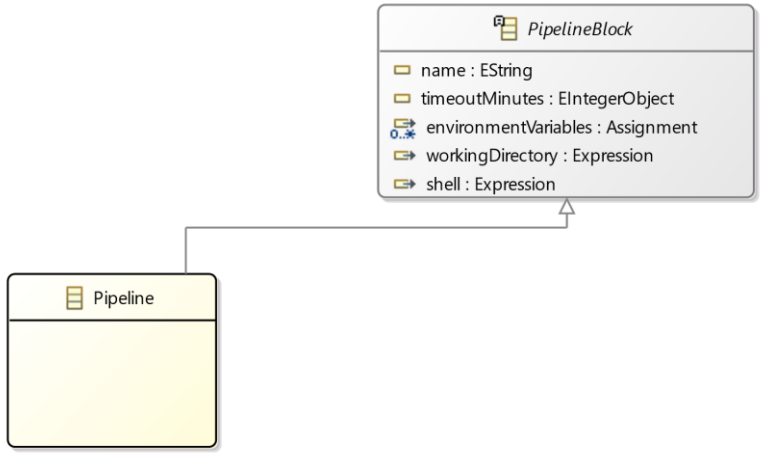
THE PLATFORM-SPECIFIC META-MODELS – ADDRESSING RQ1

What are the main core concepts shared by and unique to the different CI/CD platforms?

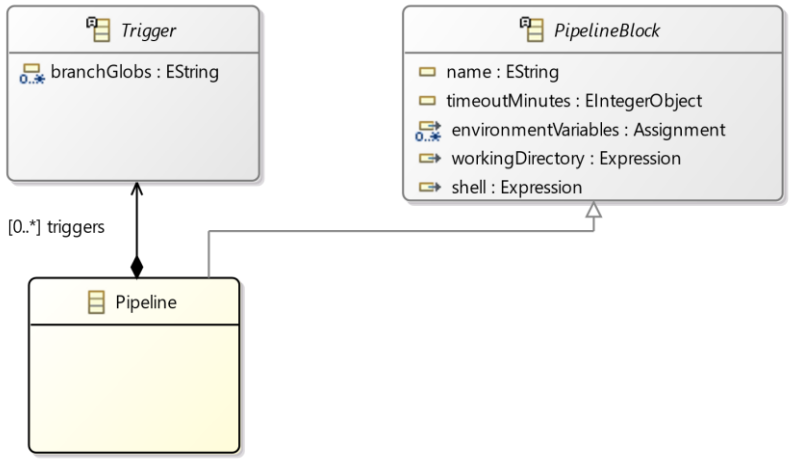
- Created platform-specific meta-models for 3 popular CI/CD platforms – GitHub Actions, CircleCI, and Jenkins.
- Close to the textual syntax of the platforms so there is no loss of concepts.
- The platforms revealed to have several core, common concepts.
- These concepts allowed the creation of a platform-independent meta-model.

THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1

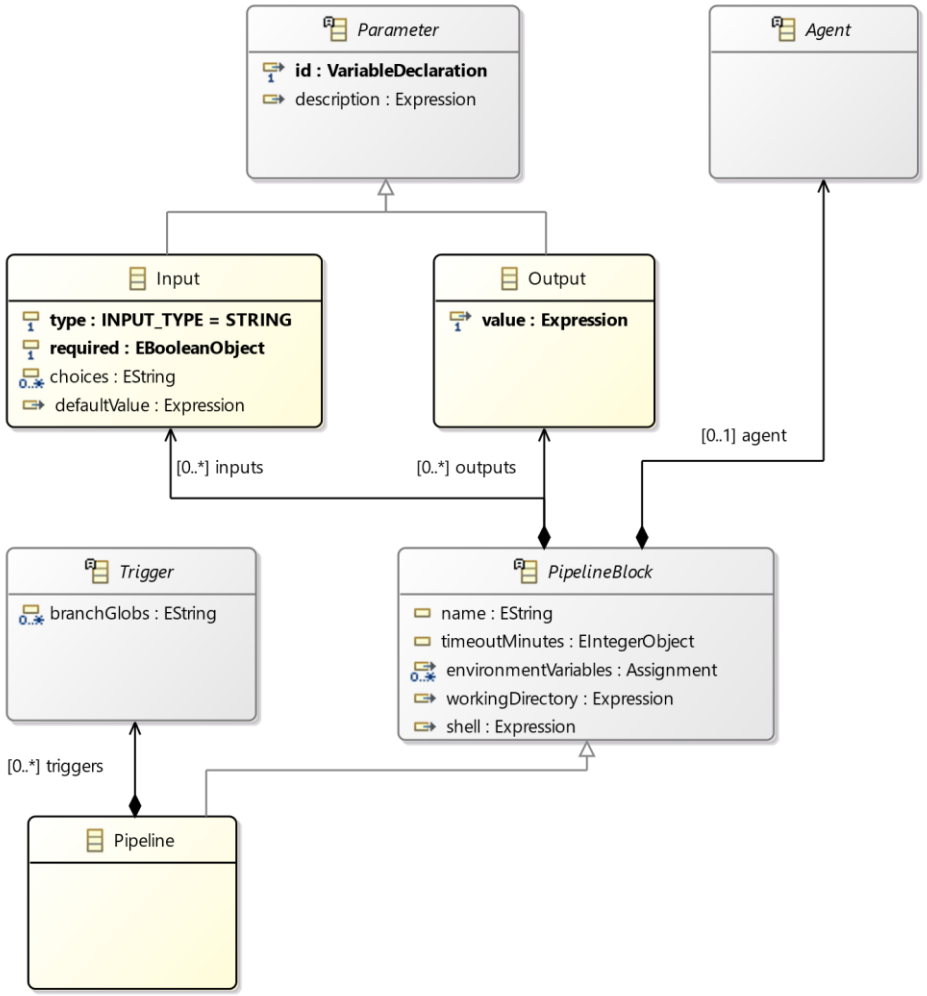
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



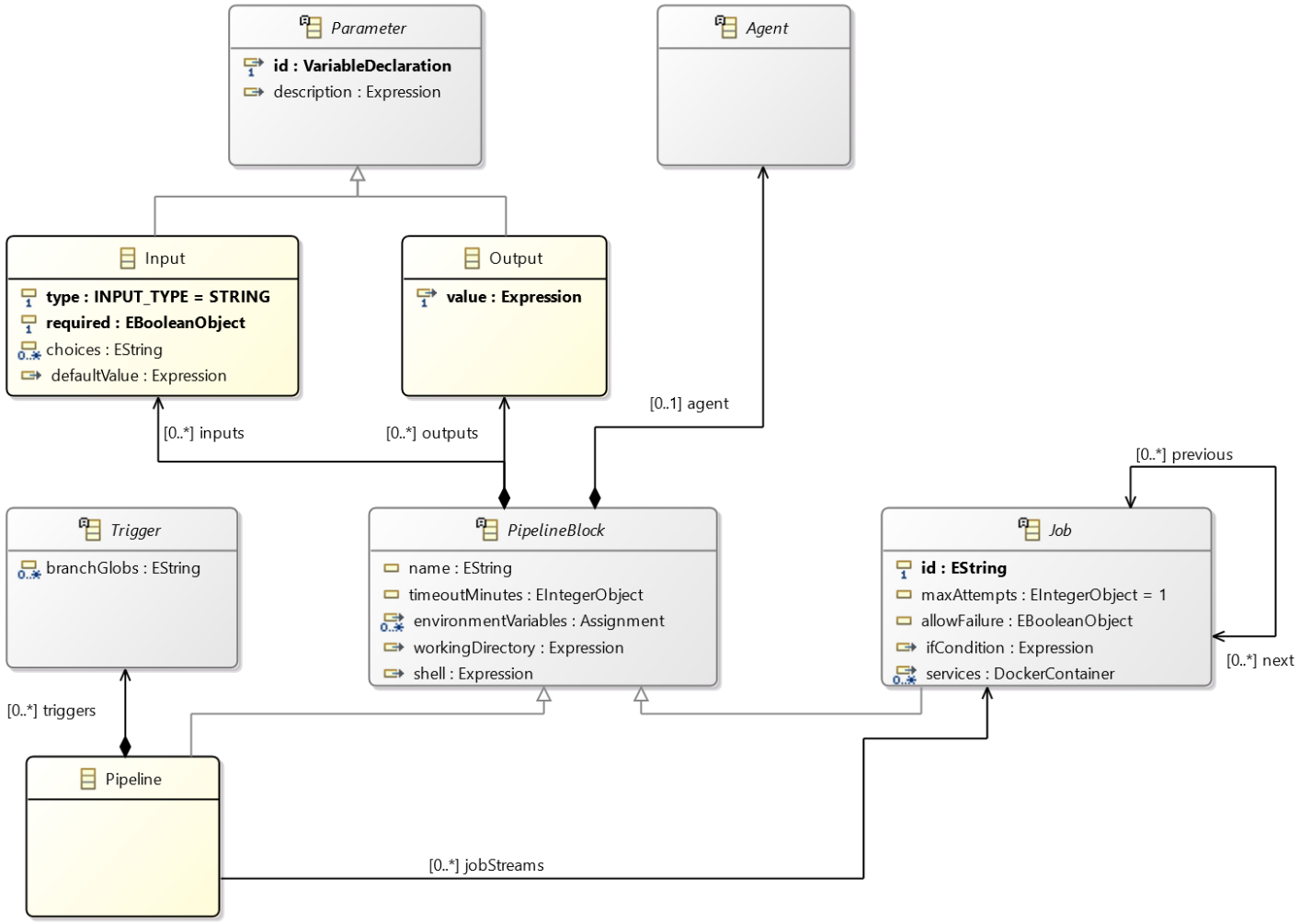
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



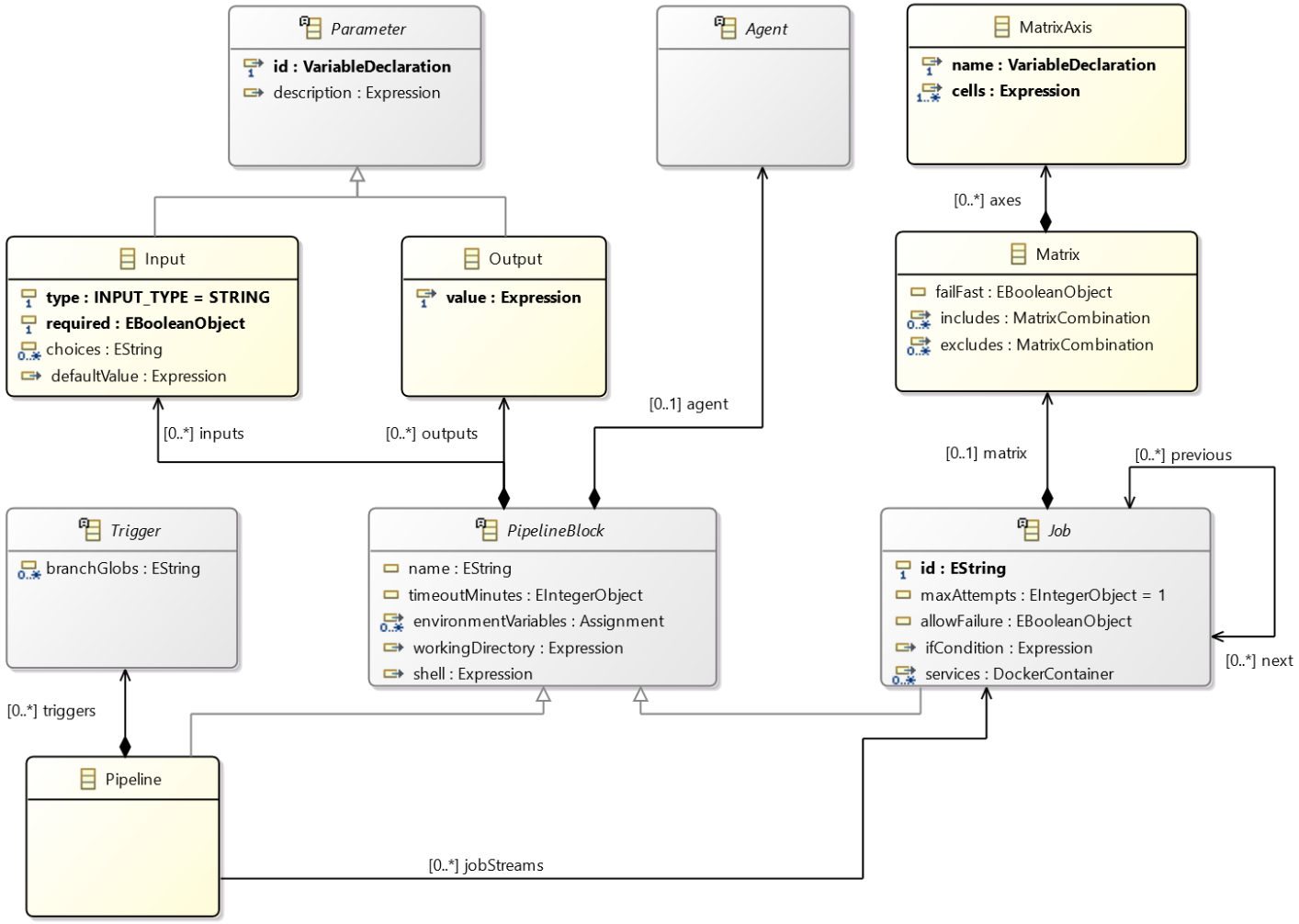
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



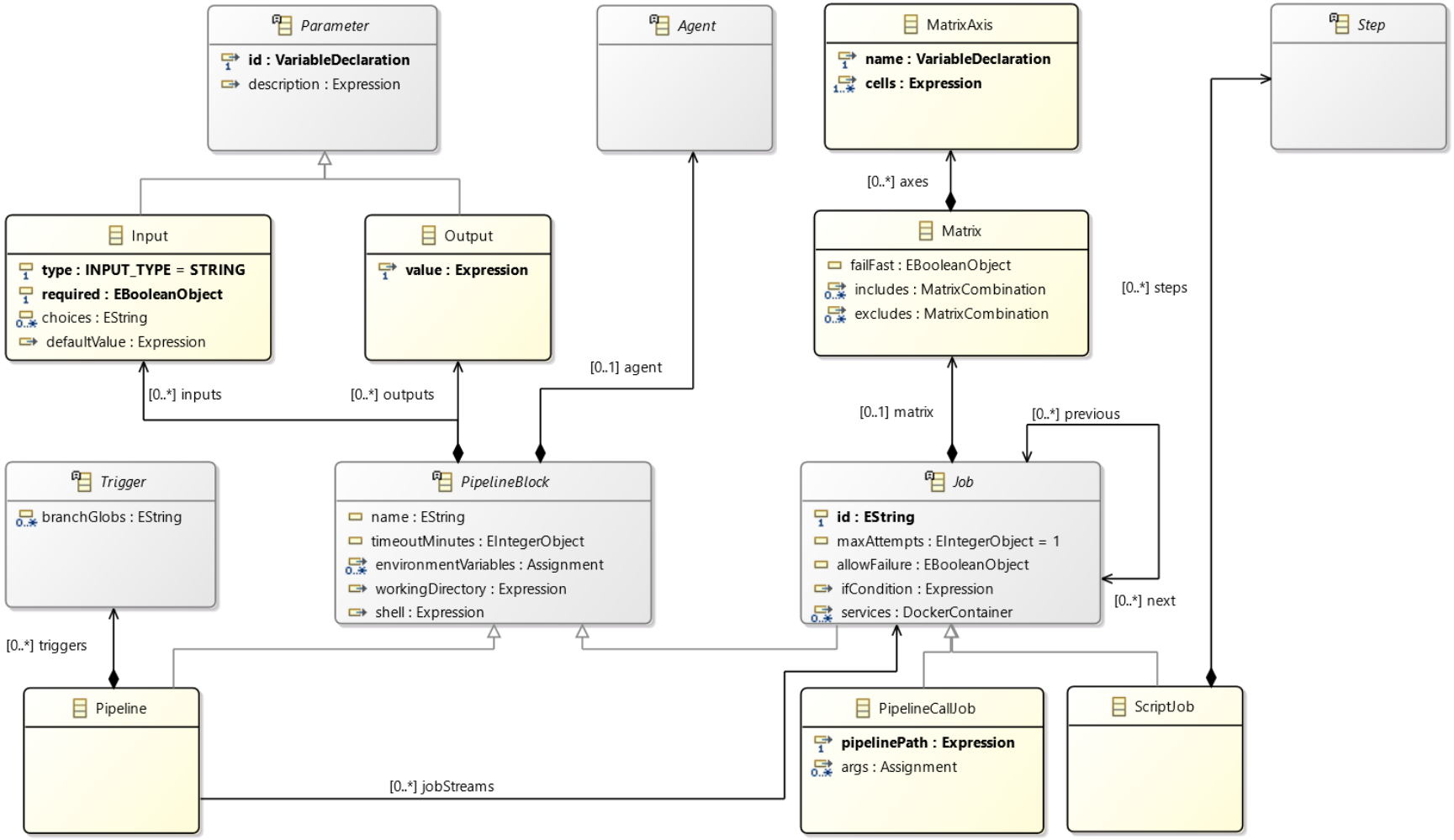
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



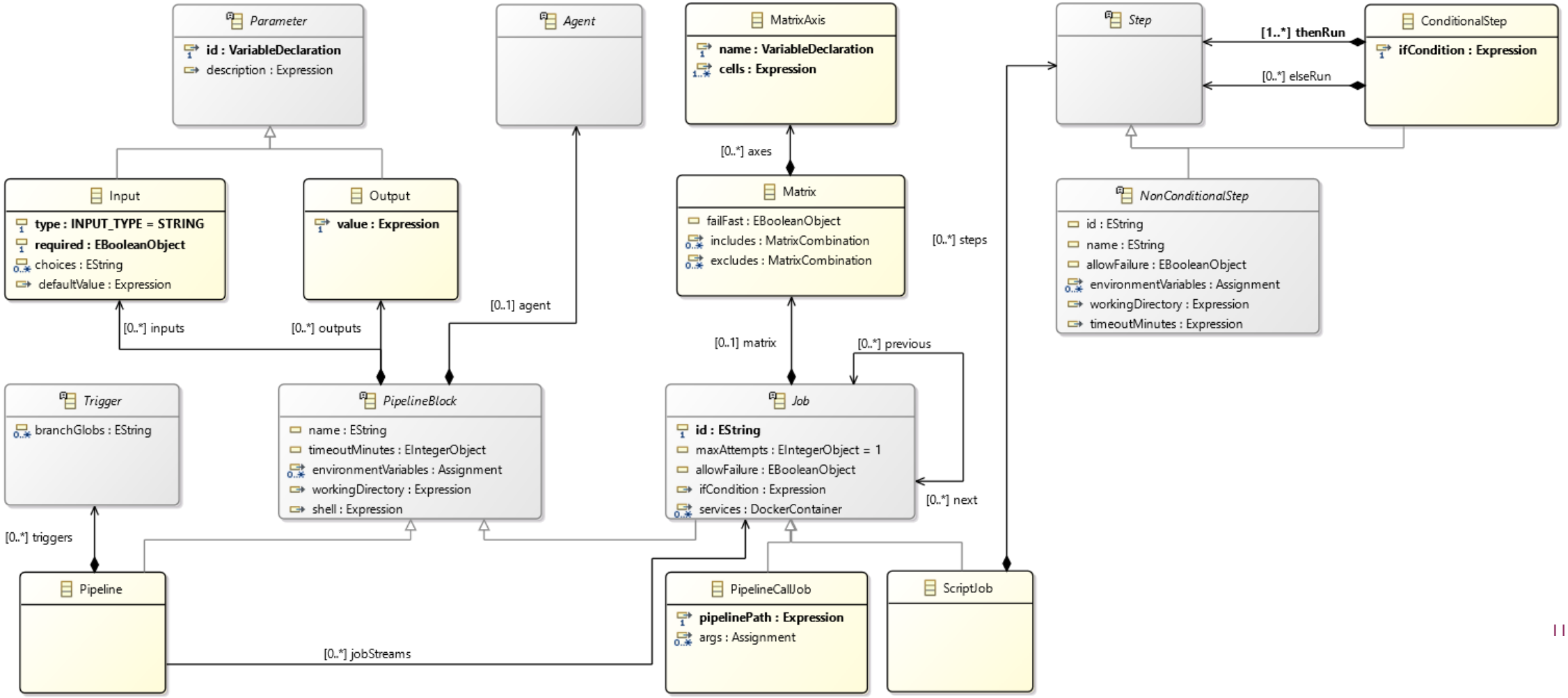
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



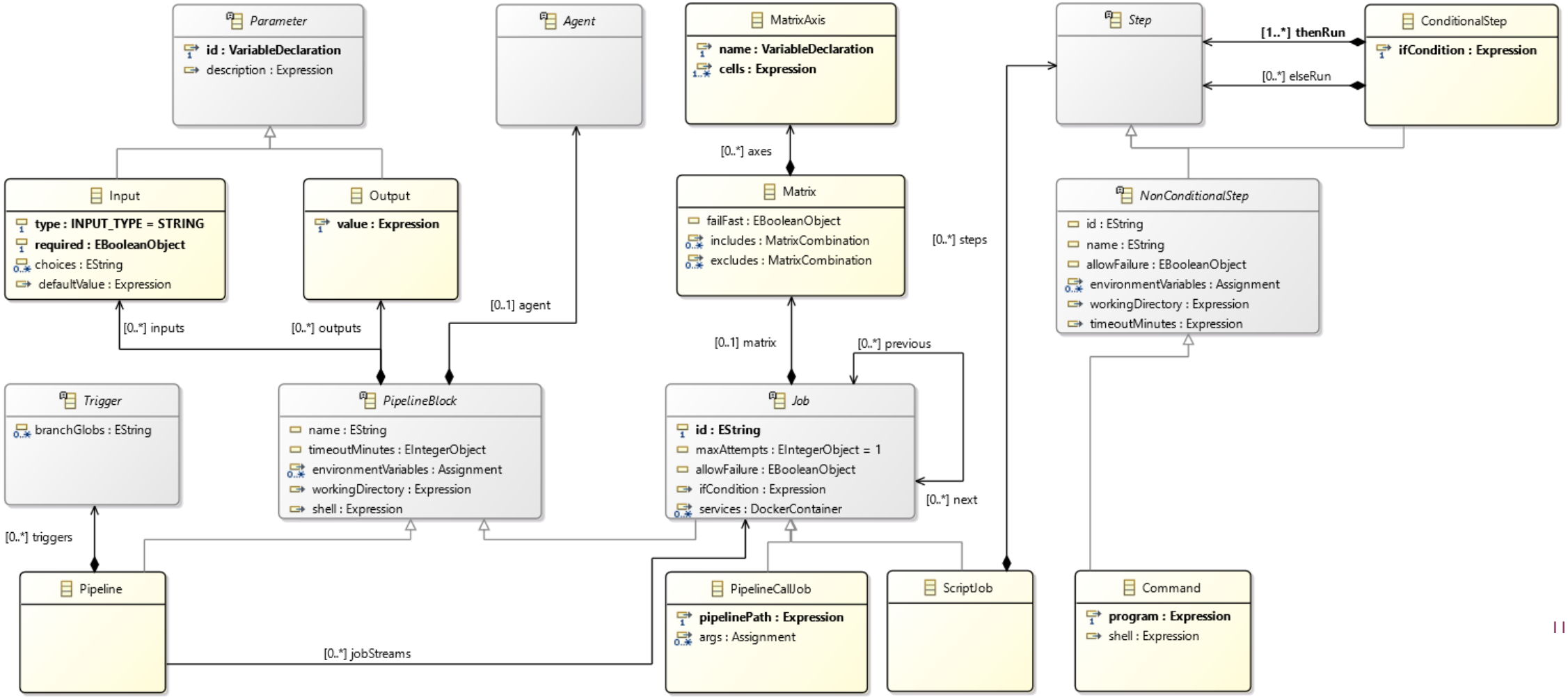
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



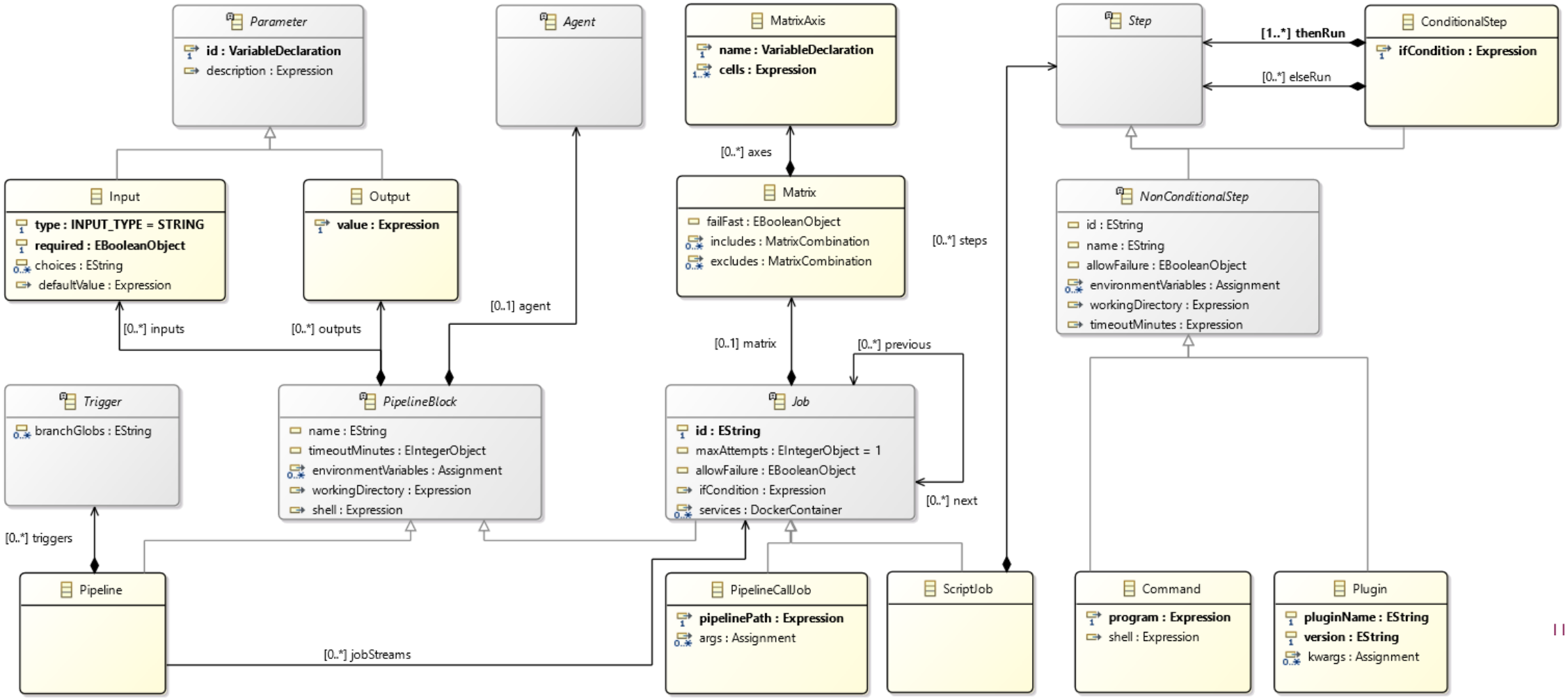
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



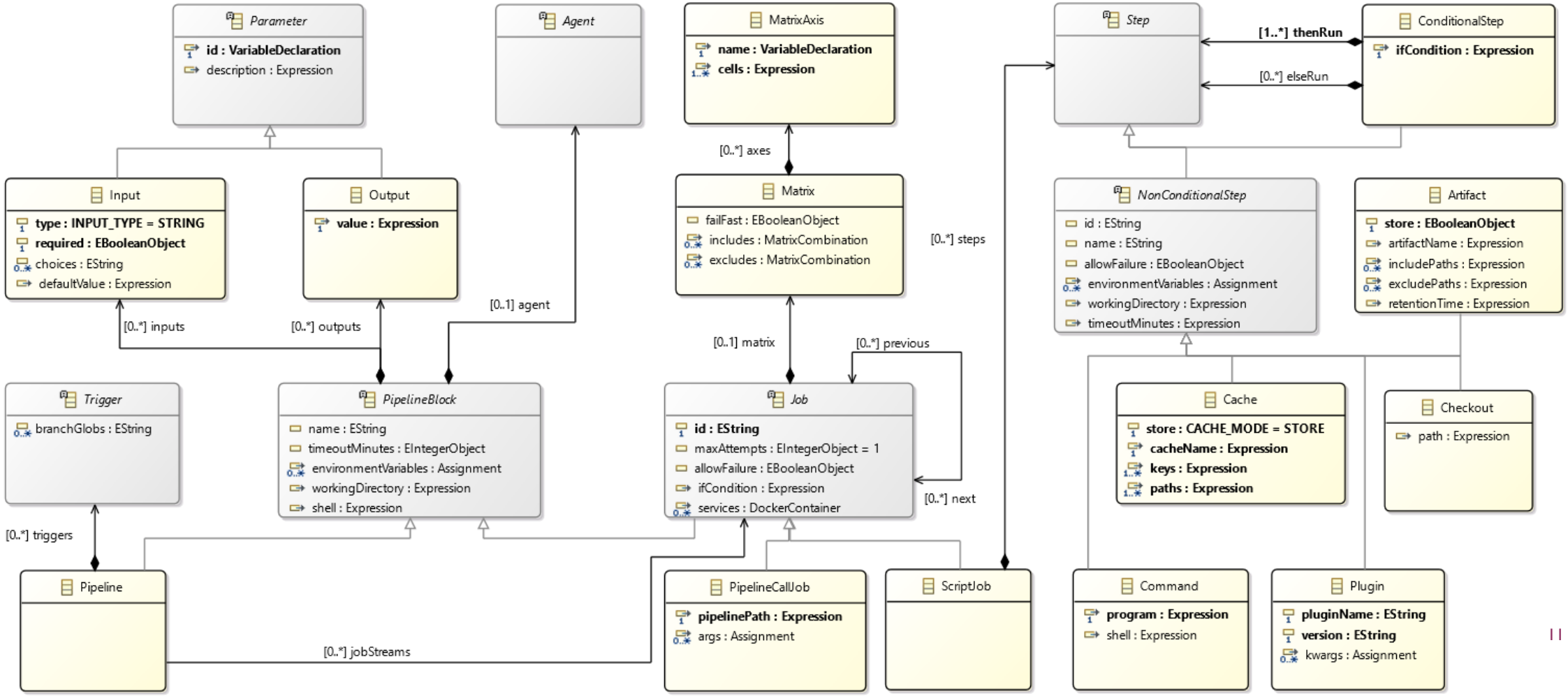
THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



THE PLATFORM-INDEPENDENT META-MODEL – ADDRESSING RQ1



THE TRANSFORMATIONS DSL

```
before translating {
    on circleci select workflow frontend
}
while translating {
    replace step 2 on 'frontend-test' with command {
        script 'npm install'
    }
}
```

THE TRANSFORMATIONS DSL

```
before translating {
    on circleci select workflow frontend
}
while translating {
    replace step 2 on 'frontend-test' with command {
        script 'npm install'
    }
}
```

- Expands migration functionality.

THE TRANSFORMATIONS DSL

```
before translating {
    on circleci select workflow frontend
}
while translating {
    replace step 2 on 'frontend-test' with command {
        script 'npm install'
    }
}
```

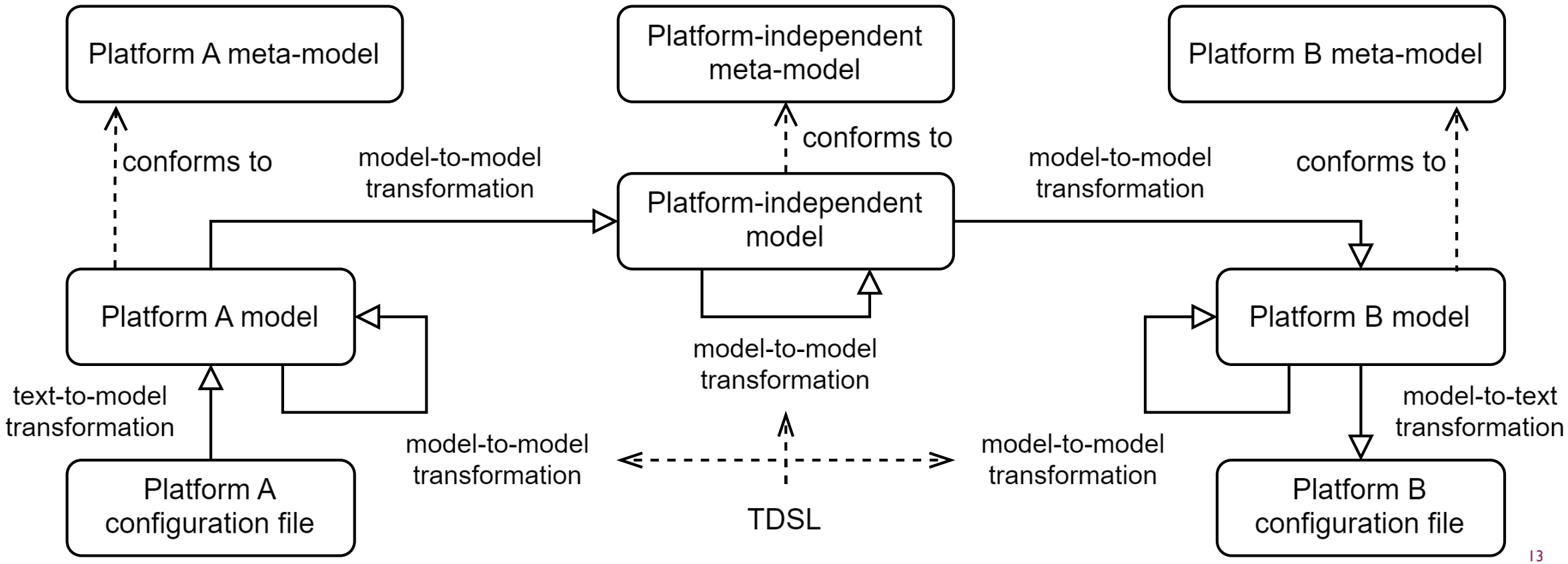
- Expands migration functionality.
- Can migrate elements that aren't migrated automatically (e.g., plugins).

THE TRANSFORMATIONS DSL

```
before translating {
    on circleci select workflow frontend
}
while translating {
    replace step 2 on 'frontend-test' with command {
        script 'npm install'
    }
}
```

- Expands migration functionality.
- Can migrate elements that aren't migrated automatically (e.g., plugins).
- By learning one syntax users can interact with many platforms.

THE TRANSFORMATIONS DSL



USING THE TRANSPILER IN PRACTICE

USING THE TRANSPILER IN PRACTICE

- The goal is to check if we can migrate pipelines using the transpiler and TDSL and keep equivalent execution.

USING THE TRANSPILER IN PRACTICE

- The goal is to check if we can migrate pipelines using the transpiler and TDSL and keep equivalent execution.
- We selected 5 example projects provided by CircleCI.

USING THE TRANSPILER IN PRACTICE

- The goal is to check if we can migrate pipelines using the transpiler and TDSL and keep equivalent execution.
- We selected 5 example projects provided by CircleCI.
- Migrated them to GitHub Actions.

USING THE TRANSPILER IN PRACTICE

- The goal is to check if we can migrate pipelines using the transpiler and TDSL and keep equivalent execution.
- We selected 5 example projects provided by CircleCI.
- Migrated them to GitHub Actions.
- Compared execution logs to determine if both pipelines have equivalent execution.

USING THE TRANSPILER IN PRACTICE

CircleCI Python example execution logs

```
(...)
Operating System: Ubuntu 20.04.6 LTS
OSType: linux
(...)
3.10.5: Pulling from cimg/python
Status: Downloaded newer image for cimg/python:3.10.5
(...)
pip install -r requirements.txt
(...)
pytest
(...)
ERROR openapi_server/test/test_cart_controller.py
ERROR openapi_server/test/test_database.py
ERROR openapi_server/test/test_image_controller.py
ERROR openapi_server/test/test_menu_controller.py
!!!!!!!!!!!!!!!!!!!! Interrupted: 4 errors during collection !
======================== 7 warnings, 4 errors in 0.64s ======
```

USING THE TRANSPILER IN PRACTICE

CircleCI Python example execution logs

```
(...)
Operating System: Ubuntu 20.04.6 LTS
OSType: linux
(...)
3.10.5: Pulling from cimg/python
Status: Downloaded newer image for cimg/python:3.10.5
(...)
pip install -r requirements.txt
(...)
pytest
(...)
ERROR openapi_server/test/test_cart_controller.py
ERROR openapi_server/test/test_database.py
ERROR openapi_server/test/test_image_controller.py
ERROR openapi_server/test/test_menu_controller.py
!!!!!!!!!!!!!!!!!!! Interrupted: 4 errors during collection !
======================== 7 warnings, 4 errors in 0.64s ======
```

GitHub Actions Python example execution logs

```
(...)
##[group]Operating System
Ubuntu
22.04.4
LTS
##[endgroup]
(...)
3.10.5: Pulling from cimg/python
Status: Downloaded newer image for cimg/python:3.10.5
(...)
pip install -r requirements.txt
(...)
pytest
(...)
ERROR openapi_server/test/test_cart_controller.py
ERROR openapi_server/test/test_database.py
ERROR openapi_server/test/test_image_controller.py
ERROR openapi_server/test/test_menu_controller.py
!!!!!!!!!!!!!!!!!!! Interrupted: 4 errors during collection !
======================== 7 warnings, 4 errors in 0.39s ======
(...)
```

USING THE TRANSPILER IN PRACTICE

USING THE TRANSPILER IN PRACTICE

- All the pipelines could be migrated while keeping equivalent execution.

USING THE TRANSPILER IN PRACTICE

- All the pipelines could be migrated while keeping equivalent execution.
- The platform-independent model supported the pipelines completely.

USING THE TRANSPILER IN PRACTICE

- All the pipelines could be migrated while keeping equivalent execution.
- The platform-independent model supported the pipelines completely.
- The TDSL was used to migrate platform-specific plugins, insert triggers, alter Docker containers, and select which CircleCI pipeline to migrate.

USING THE TRANSPILER IN PRACTICE

- All the pipelines could be migrated while keeping equivalent execution.
- The platform-independent model supported the pipelines completely.
- The TDSL was used to migrate platform-specific plugins, insert triggers, alter Docker containers, and select which CircleCI pipeline to migrate.
- All TDSL transformations were executed on the platform-independent model (except selecting the CircleCI pipelines)

EVALUATING THE TRANSPILER WITH MANY PIPELINES

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- The goal is to determine transpiler limitations by migrating a larger number of pipelines.

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- The goal is to determine transpiler limitations by migrating a larger number of pipelines.
- We migrate pipelines twice. Firstly, from the original to an intermediary platform and then back to the original platform.

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- The goal is to determine transpiler limitations by migrating a larger number of pipelines.
- We migrate pipelines twice. Firstly, from the original to an intermediary platform and then back to the original platform.
- Compare original and migrated scripts and determine if they are semantically equivalent.

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- The goal is to determine transpiler limitations by migrating a larger number of pipelines.
- We migrate pipelines twice. Firstly, from the original to an intermediary platform and then back to the original platform.
- Compare original and migrated scripts and determine if they are semantically equivalent.
- Before migration, run a validation to check if the transpiler currently supports it fully.

EVALUATING THE TRANSPILER WITH MANY PIPELINES

Original pipeline (GitHub Actions)

```yaml
name: "Workflow"

on: push

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
```

Intermediate pipeline (CircleCI)

```yaml
version: 2.1

jobs:
  build:
    machine:
      image: "ubuntu-latest"
    steps:
      - checkout:

workflows:
  version: 2.1
  Workflow:
    jobs:
      -
        build:
```

Migrated pipeline (GitHub Actions)

```yaml
name: "Workflow"

jobs:
  build:
    name: "build"
    runs-on:
    - "ubuntu-latest"
    steps:
    - uses: "actions/checkout@v4"
```

EVALUATING THE TRANSPILER WITH MANY PIPELINES

Original pipeline (GitHub Actions)

```yaml
name: "Workflow"

on: push

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
```

Intermediate pipeline (CircleCI)

```yaml
version: 2.1

jobs:
  build:
    machine:
      image: "ubuntu-latest"
    steps:
      - checkout:

workflows:
  version: 2.1
  Workflow:
    jobs:
      -
        build:
```

Migrated pipeline (GitHub Actions)

```yaml
name: "Workflow"

jobs:
  build:
    name: "build"
    runs-on:
    - "ubuntu-latest"
    steps:
    - uses: "actions/checkout@v4"
```

EVALUATING THE TRANSPILER WITH MANY PIPELINES

Original pipeline (GitHub Actions)

```yaml
name: "Workflow"

on: push

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
```

Intermediate pipeline (CircleCI)

```yaml
version: 2.1

jobs:
  build:
    machine:
      image: "ubuntu-latest"
    steps:
      - checkout:

workflows:
  version: 2.1
  Workflow:
    jobs:
      -
        build:
```

Migrated pipeline (GitHub Actions)

```yaml
name: "Workflow"

jobs:
  build:
    name: "build"
    runs-on:
    - "ubuntu-latest"
    steps:
    - uses: "actions/checkout@v4"
```

EVALUATING THE TRANSPILER WITH MANY PIPELINES

Original pipeline (GitHub Actions)

```yaml
name: "Workflow"

on: push

jobs:
  build:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v3
```

Intermediate pipeline (CircleCI)

```yaml
version: 2.1

jobs:
  build:
    machine:
      image: "ubuntu-latest"
    steps:
      - checkout:

workflows:
  version: 2.1
  Workflow:
    jobs:
      -
        build:
```

Migrated pipeline (GitHub Actions)

```yaml
name: "Workflow"

jobs:
  build:
    name: "build"
    runs-on:
    - "ubuntu-latest"
    steps:
    - uses: "actions/checkout@v4"
```

EVALUATING THE TRANSPILER WITH MANY PIPELINES

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- Evaluated 25,487 scripts from 10,000 repositories using GitHub Actions

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- Evaluated 25,487 scripts from 10,000 repositories using GitHub Actions
- The transpiler supported CircleCI migration for 4,091 scripts.

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- Evaluated 25,487 scripts from 10,000 repositories using GitHub Actions
- The transpiler supported CircleCI migration for 4,091 scripts.
- Most scripts that failed validation (82,3%) were due to references to variables not declared in the pipeline (e.g., API tokens, commit SHA).

EVALUATING THE TRANSPILER WITH MANY PIPELINES

- Evaluated 25,487 scripts from 10,000 repositories using GitHub Actions
- The transpiler supported CircleCI migration for 4,091 scripts.
- Most scripts that failed validation (82,3%) were due to references to variables not declared in the pipeline (e.g., API tokens, commit SHA).
- 81,1% of the supported scripts were migrated without semantic change.

DISCUSSION – ADDRESSING RQ2

DISCUSSION – ADDRESSING RQ2

Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

DISCUSSION – ADDRESSING RQ2

Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

- Yes.

DISCUSSION – ADDRESSING RQ2

Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

- Yes.
- In the first evaluation, the platform-independent meta-model supported the pipelines completely, all but one of the TDSL transformations were done on the platform-independent model.

DISCUSSION – ADDRESSING RQ2

Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

- Yes.
- In the first evaluation, the platform-independent meta-model supported the pipelines completely, all but one of the TDSL transformations were done on the platform-independent model.
- In the second evaluation, while many pipelines are not supported yet, most supported pipelines can be migrated without semantic alteration.

DISCUSSION – ADDRESSING RQ2

Can a platform-independent meta-model be the basis for the accurate translation of CI/CD pipelines between platforms?

- Yes.
- In the first evaluation, the platform-independent meta-model supported the pipelines completely, all but one of the TDSL transformations were done on the platform-independent model.
- In the second evaluation, while many pipelines are not supported yet, most supported pipelines can be migrated without semantic alteration.
- Abstraction to the platform-independent meta-model will always result in alterations in some cases.

DISCUSSION – ADDRESSING RQ3

DISCUSSION – ADDRESSING RQ3

Can CI/CD pipeline migration be fully automated?

DISCUSSION – ADDRESSING RQ3

Can CI/CD pipeline migration be fully automated?

- No.

DISCUSSION – ADDRESSING RQ3

Can CI/CD pipeline migration be fully automated?

- No.
- Platforms have different feature sets. These are one of the main motivators for migration.

DISCUSSION – ADDRESSING RQ3

Can CI/CD pipeline migration be fully automated?

- No.
- Platforms have different feature sets. These are one of the main motivators for migration.
- Migrating CI/CD sometimes necessitates changes to the codebase or changes to the pipeline that require context-specific knowledge.

DISCUSSION – ADDRESSING RQ3

Can CI/CD pipeline migration be fully automated?

- No.
- Platforms have different feature sets. These are one of the main motivators for migration.
- Migrating CI/CD sometimes necessitates changes to the codebase or changes to the pipeline that require context-specific knowledge.
- Plugins need to be changed between platforms, there is no guarantee there will always be a corresponding plugin in another platform.

CONCLUSIONS AND FUTURE WORK

CONCLUSIONS AND FUTURE WORK

- The enough common concepts to CI/CD platforms to define a common language.

CONCLUSIONS AND FUTURE WORK

- The enough common concepts to CI/CD platforms to define a common language.
- Migration requires some manual work.

CONCLUSIONS AND FUTURE WORK

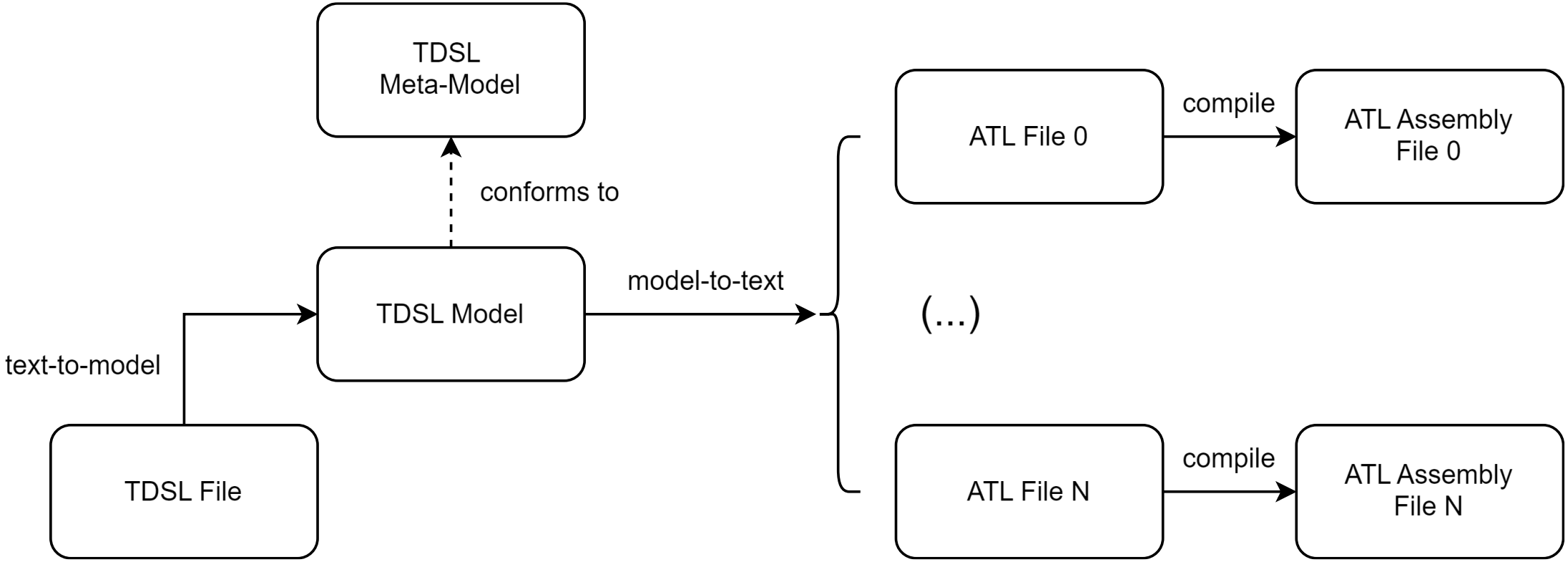
- The enough common concepts to CI/CD platforms to define a common language.
- Migration requires some manual work.
- Future versions of the TDSL could create a *lingua franca* for CI/CD pipelines.

CONCLUSIONS AND FUTURE WORK

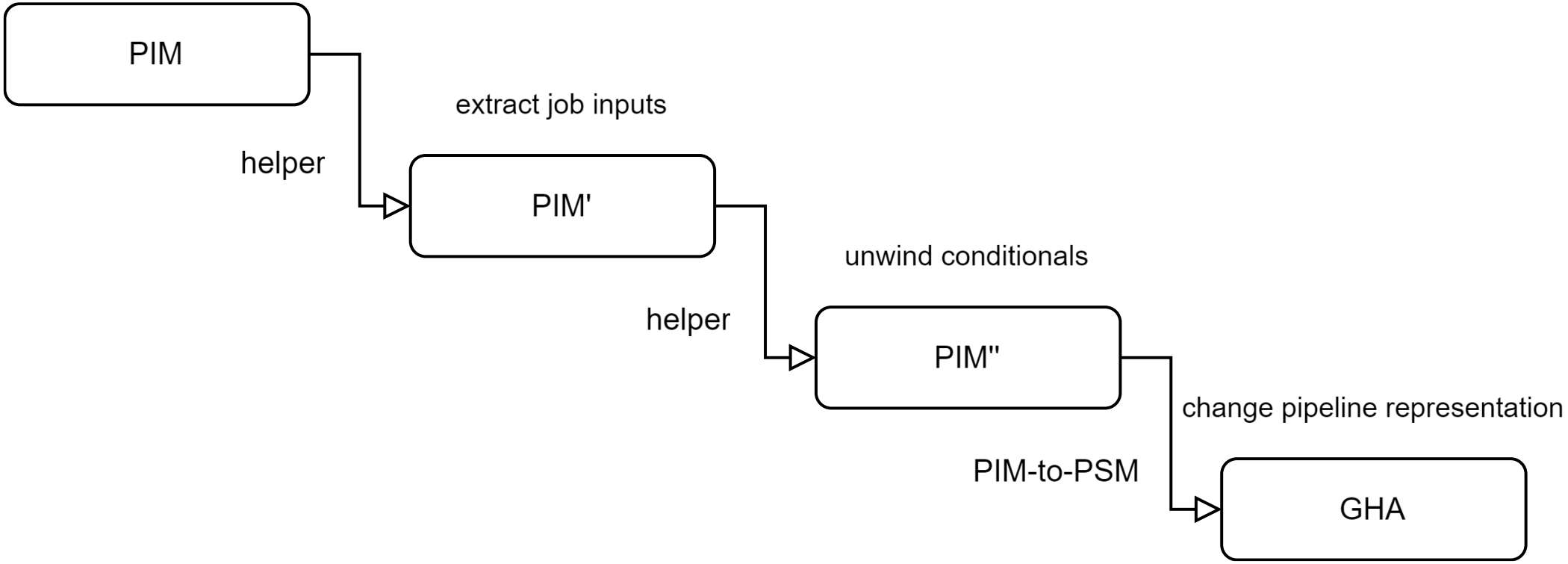
- The enough common concepts to CI/CD platforms to define a common language.
- Migration requires some manual work.
- Future versions of the TDSL could create a *lingua franca* for CI/CD pipelines.
- There is room for further development of the platform-independent meta-model.

THANK YOU!

TDSL REENGINEERING PROCESS



PIM-TO-PSM COMPOUND TRANSFORMATION



DOUBLE ROUND-TRIP CHANGES

Of 775 pipelines with semantic change (pipelines may have multiple changes):

- 404 had **Plugins** lose arguments when being migrated to **Checkouts**, **Artifacts**, or **Caches**. This is because they have extra functionality not supported by the PIMM.
- 31 had lost **Plugin** environment variables. CircleCI does not natively support environment variables in **Orb** steps. We send these as arguments instead. This avoids loss of information as, when changing the GHA **Plugin** to a CircleCI one, the CircleCI one may instead take these values as arguments.
- 100 had differences because strings were parsed as floating point numbers. This happens most in **Plugins** as we have no information on the type of the argument we are parsing. The string value "3.10" is parsed as a float 3.1. This causes changes mostly when the **Plugin** argument indicates a version of some kind, as 3.10 should be read as a string in that context.
- 16 had differences due to encoding. The transpiler only supports UTF-8.
- 54 had macOS version mismatches as CircleCI does not directly store the macOS version.
- 252 had differences that are not easily classifiable. These should be seen as the result of bugs in the current version of the transpiler.

COMPLEX TDSL EXAMPLE

```
while {
    add trigger when "input.triggers->isEmpty()" manual
    set container image when 'true' to 'node:16'
    replace step 2 on 'test' with command {
        script 'yarn install'
    }
    insert step 1 on 'cypress/run' with checkout  {}
    replace step 2 on 'cypress/run' with plugin {
        name 'cypress-io/github-action'
        version 'v6'
        args {
            'command' = 'yarn  run test:e2e --headless'
        }
    }
    run atl on cicd {
    "
-- @path CICD=/d.fe.up.pt.cicd.metamodel/model/CICD.ecore

module cicdRefinement;
create OUT : CICD refining IN : CICD;

rule RemoveContainer {
    from
        input : CICD!DockerContainer (
            input.refImmediateComposite().refImmediateComposite().id =
            ↪ 'cypress/run'
        )
    to
        drop
}
    "
    }
}
```

REFERENCES

1. K. Beck. "Embracing change with extreme programming". In: Computer 32.10 (Oct.1999), pp. 70–77. ISSN: 00189162. DOI: 10.1109/2.796139. URL: http://ieeexplore.ieee.org/document/796139/ (visited on 12/09/2023).
2. Jez Humble and David Farley. Continuous Delivery: Reliable Software Releases through Build, Test, and Deployment Automation. Pearson Education, July 27, 2010. 956 pp. ISBN: 978-0-321-67022-9.
3. Hugo Gião, André Flores, Jácome Cunha, and Rui Pereira. Chronicles of CI/CD: A Deep Dive into its Usage Over Time. Manuscript submitted for publication. 2023.
4. Pooya Rostami Mazrae et al. "On the usage, co-usage and migration of CI/CD tools: A qualitative analysis". In: Empirical Software Engineering 28.2 (Mar. 7, 2023), p. 52. DOI: 10.1007/s10664-022-10285-5. URL: https://doi.org/10.1007/s10664-022-10285-5.
5. Introduction to CircleCI migration - CircleCI. URL: https://circleci.com/docs/migration-intro/ (visited on 12/06/2023).

REFERENCES

6. Marco Brambilla, Jordi Cabot, and Manuel Wimmer. Model-Driven Software Engineering in Practice. Synthesis Lectures on Software Engineering. Cham: Springer International Publishing, 2017. ISBN: 978-3-031-01421-5. DOI: 10.1007/978-3-031-02549-5. URL: https://link.springer.com/10.1007/978-3-031-02549-5 (visited on 12/19/2023).
7. Automating migration with GitHub Actions Importer. GitHub Docs. URL: https://docs.github.com/en/actions/migrating-to-github-actions/automated-migrations/automating-migration-with-github-actions-importer (visited on 12/06/2023).
8. Hugo Gião, Jácome Cunha, and Rui Pereira. Model-Driven Approaches for DevOps: A Systematic Literature Review. Manuscript submitted for publication. 2023.
9. Alessandro Colantoni, Luca Berardinelli, and Manuel Wimmer. "DevOpsML: towards modeling DevOps processes and platforms". In: Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. MODELS '20: ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems. Virtual Event Canada: ACM, Oct. 16, 2020, pp. 1–10. ISBN: 978-1-4503-8135-2. DOI: 10.1145/3417990.3420203. URL: https://dl.acm.org/doi/10.1145/3417990.3420203 (visited on 12/07/2023).

REFERENCES

10. Rakesh Kumar and Rinkaj Goyal. "Modeling continuous security: A conceptual model for automated DevSecOps using open-source software over cloud (ADOC)". In: Computers & Security 97 (Oct. 1, 2020), p. 101967. ISSN: 0167-4048. DOI: 10.1016/j.cose.2020.101967. URL: https://www.sciencedirect.com/science/article/pii/S0167404820302406 (visited on 01/03/2024).
11. Nicolas Ferry et al. "ENACT: Development, Operation, and Quality Assurance of Trustworthy Smart IoT Systems". In: Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment. Ed. by Jean-Michel Bruel, Manuel Mazzara, and Bertrand Meyer. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2019, pp. 112–127. ISBN: 978-3-030-06019-0. DOI: 10.1007/978-3-030-06019-0_9.
12. Badr El Khalyly et al. "A new metamodel approach of CI/CD applied to Internet of Things Ecosystem". In: 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS). 2020 IEEE 2nd International Conference on Electronics, Control, Optimization and Computer Science (ICECOCS). Kenitra, Morocco: IEEE, Dec. 2, 2020, pp. 1–6. ISBN: 978-1-72816-921-7. DOI: 10.1109/ICECOCS50124 . 2020 . 9314485. URL: https://ieeexplore.ieee.org/document/9314485/ (visited on 12/07/2023).

REFERENCES

13. OASIS. Topology and orchestration specification for cloud applications (TOSCA) Version 1.0, Committee Specification 01. URL: http://docs.oasisopen.org/tosca/TOSCA/v1.0/cs01/TOSCA-v1.0-cs01.html. (visited on 01/04/2024).
14. Johannes Wettinger et al. "Streamlining DevOps automation for Cloud applications using TOSCA as standardized metamodel". In: Future Generation Computer Systems 56 (Mar. 2016), pp. 317–332. ISSN: 0167739X. DOI: 10.1016/j.future.2015.07.017. URL: https://linkinghub.elsevier.com/retrieve/pii/S0167739X15002496 (visited on 01/04/2024).
15. Elisabetta Di Nitto et al., eds. Model-Driven Development and Operation of Multi-Cloud Applications: The MODAClouds Approach. SpringerBriefs in Applied Sciences and Technology. Cham: Springer International Publishing, 2017. ISBN: 978-3-319-46031-4. DOI: 10.1007/978-3-319-46031-4. URL: http: //link.springer.com/10.1007/978-3-319-46031-4 (visited on 01/04/2024).
16. Geir Horn and Pawel Skrzypek. "MELODIC: Utility Based Cross Cloud Deployment Optimisation". In: 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA). 2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA). Krakow: IEEE, May 2018, pp. 360–367. ISBN: 978-1-5386-5395-1. DOI: 10.1109/ WAINA. 2018.00112. URL: https://ieeexplore.ieee.org/document/8418097/ (visited on 01/04/2024).

REFERENCES

17. Wei Chen et al. "MORE: A Model-Driven Operation Service for Cloud-Based IT Systems". In: 2016 IEEE International Conference on Services Computing (SCC). 2016 IEEE International Conference on Services Computing (SCC). San Francisco, CA, USA: IEEE, June 2016, pp. 633–640. ISBN: 978-1-5090-2628-9. DOI: 10.1109/SCC.2016.88. URL: http://ieeexplore.ieee.org/document/7557508/ (visited on 01/04/2024).
18. Zia Babar, Alexei Lapouchnian, and Eric Yu. "Modeling DevOps Deployment Choices Using Process Architecture Design Dimensions". In: The Practice of Enterprise Modeling. Ed. by Jolita Ralyté, Sergio España, and Óscar Pastor. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, 2015, pp. 322–337. ISBN: 978-3-319-25897-3. DOI: 10.1007/978-3-319-25897-3_21.
19. Michael Wurster et al. "The essential deployment metamodel: a systematic review of deployment automation technologies". In: SICS Software-Intensive Cyber-Physical Systems 35.1 (Aug. 2020), pp. 63–75. ISSN: 2524-8510, 2524-8529. DOI: 10.1007/s00450-019-00412-x. URL: http://link.springer.com/10.1007/s00450-019-00412-x (visited on 01/04/2024).
20. Fran Melchor et al. "A Model-Driven Approach for Systematic Reproducibility and Replicability of Data Science Projects". In: Advanced Information Systems Engineering. Ed. By Xavier Franch et al. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2022, pp. 147–163. ISBN: 978-3-031-07472-1. DOI: 10.1007/978-3-031-07472-1_9.

REFERENCES

21. Julio Sandobalin. "A Model-Driven Approach to Continuous Delivery of Cloud Resources". In: Service-Oriented Computing – ICSOC 2017 Workshops. Ed. by Lars Braubach et al. Vol. 10797. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 346–351. ISBN: 978-3-319-91763-4. DOI: 10.1007/978-3-319-91764-1_29. URL: https://link.springer.com/10.1007/978-3-319-91764-1_29 (visited on 12/07/2023).
22. Willem-Jan van den Heuvel et al. "ChainOps for Smart Contract-Based Distributed Applications". In: Business Modeling and Software Design. Ed. by Boris Shishkov. Lecture Notes in Business Information Processing. Cham: Springer International Publishing, 2021, pp. 374–383. ISBN: 978-3-030-79976-2. DOI: 10.1007/978-3-030-79976-2_25.
23. Jörg Christian Kirchhof et al. "MontiThings: Model-Driven Development and Deployment of Reliable IoT Applications". In: Journal of Systems and Software 183 (Jan. 2022), p. 111087. ISSN: 01641212. DOI: 10.1016/j.jss.2021.111087. URL: https://linkinghub.elsevier.com/retrieve/pii/S0164121221001849 (visited on 12/07/2023).
24. Hui Song et al. "Model-based fleet deployment in the IoT–edge–cloud continuum". In: Software and Systems Modeling 21.5 (Oct. 2022), pp. 1931–1956. ISSN: 1619-1366, 1619-1374. DOI: 10.1007/s10270-022-01006-z. URL: https://link.springer.com/10.1007/s10270-022-01006-z (visited on 12/07/2023).

REFERENCES

25. Jerome Hugues et al. "TwinOps - DevOps meets model-based engineering and digital twins for the engineering of CPS". In: Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. MODELS '20: ACM/IEEE 23rd International Conference on Model Driven Engineering Languages and Systems. Virtual Event Canada: ACM, Oct. 16, 2020, pp. 1–5. ISBN: 978-1-4503-8135-2. DOI: 10.1145/3417990.3421446. URL: https://dl.acm.org/doi/10.1145/3417990.3421446 (visited on 12/07/2023).
26. Alessandro Colantoni et al. "Towards blended modeling and simulation of DevOps processes: the Keptn case study". In: Proceedings of the 25th International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings. MODELS'22: ACM/IEEE 25th International Conference on Model Driven Engineering Languages and Systems. Montreal Quebec Canada: ACM, Oct. 23, 2022, pp. 784–792. ISBN: 978-1-4503-9467-3. DOI: 10.1145/3550356.3561597. URL: https://dl.acm.org/doi/10.1145/3550356.3561597 (visited on 12/07/2023).
27. Bart Meyers et al. "A Model-Driven Engineering Framework to Support the Functional Safety Process". In: 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C). 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C). Sept. 2019, pp. 619–623. DOI: 10.1109/MODELS-C.2019.00094. URL: https://ieeexplore.ieee.org/abstract/document/8904799 (visited on 01/03/2024).

REFERENCES

28. Luis F. Rivera et al. "UML-driven automated software deployment". In: Proceedings of the 28th Annual International Conference on Computer Science and Software Engineering. CASCON '18. USA: IBM Corp., Oct. 29, 2018, pp. 257–268. (Visited on 01/03/2024).
29. Franklin Magalhães Ribeiro et al. "A Model-Driven Solution for Automatic Software Deployment in the Cloud". In: Information Technology: New Generations. Ed. by Shahram Latifi. Advances in Intelligent Systems and Computing. Cham: Springer International Publishing, 2016, pp. 591–601. ISBN: 978-3-319-32467-8. DOI: 10.1007/978-3-319-32467-8_52.
30. Antonio Bucchiarone, Antonio Cicchetti, and Annapaola Marconi. "Exploiting Multilevel Modelling for Designing and Deploying Gameful Systems". In: 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS). 2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS). Munich, Germany: IEEE, Sept. 2019, pp. 34–44. ISBN: 978-1-72812-536-7. DOI: 10.1109/MODELS.2019.00-17. URL: https://ieeexplore.ieee.org/document/8906924/ (visited on 01/04/2024).
31. Matej Artac et al. "Model-driven continuous deployment for quality DevOps". In: Proceedings of the 2nd International Workshop on Quality-Aware DevOps. ISSTA '16: International Symposium on Software Testing and Analysis. Saarbrücken Germany: ACM, July 21, 2016, pp. 40–41. ISBN: 978-1-4503-4411-1. DOI: 10.1145/2945408.2945417. URL: https://dl.acm.org/doi/10.1145/2945408.2945417 (visited on 01/04/2024).

REFERENCES

32. Abdelmadjid Ketfi and Noureddine Belkhatir. "Model-driven framework for dynamic deployment and reconfiguration of component-based software systems". In: Proceedings of the 2005 symposia on Metainformatics - MIS '05. the 2005 symposia. Esbjerg, Denmark: ACM Press, 2005, 8–es. ISBN: 978-1-59593-719-3. DOI: 10 .1145/1234324.1234332. URL: http://portal.acm.org/citation.cfm?doid=1234324.1234332 (visited on 01/04/2024).
33. Ana C. Franco Da Silva et al. "OpenTOSCA for IoT: Automating the Deployment of IoT Applications based on the Mosquitto Message Broker". In: Proceedings of the 6th International Conference on the Internet of Things. IoT'16: The 6th International Conference on the Internet of Things. Stuttgart Germany: ACM, Nov. 7, 2016, pp. 181–182. ISBN: 978-1-4503-4814-0. DOI: 10.1145/2991561.2998464. URL: https://dl.acm.org/doi/10.1145/2991561.2998464 (visited on 01/04/2024).