CIM to PIM Transformation: An Analytical Survey

Hamid Reza Sharifi
Department of Computer Engineering
Science and Research Branch,
Islamic Azad University
Tehran, Iran
hr.sharifi@srbiau.ac.ir

Mehran Mohsenzadeh
Department of Computer Engineering
Science and Research Branch,
Islamic Azad University
Tehran, Iran
mohsenzadeh@srbiau.ac.ir

Seyyed Mohsen Hashemi
Department of Computer Engineering
Science and Research Branch,
Islamic Azad University
Tehran, Iran
hashemi@srbiau.ac.ir

Abstract
Model transformation is one of the main issues and key elements in Model Driven Architecture (MDA). Computation Independent Model (CIM) to Platform Independent Model (PIM) transformation is the first transformation in MDA and it is very important in designing high quality software. Several CIM to PIM transformation approaches have emerged, they have not been objectively analysed yet. In this paper, we provide a review of several CIM to PIM transformation approaches, and present a criteria-based evaluation. The results can be used for assessing and comparing CIM to PIM transformation approaches.

Keywords: MDA; CIM; PIM; Transformation; Evaluation Criteria.

1. Introduction

The MDA is an approach for software development defined by the Object Management Group (OMG). MDA defines four models in various levels of abstraction [1]. They are CIM, PIM, PSM, and Code.

CIM is a model that describes a system from the computation independent viewpoint. PIM is a model that contains no information specific to the platform. Platform Specific Model (PSM) is a model that includes information about the specific technology that is used in the realization of it on a specific platform. Code is a specification of the system in the source code. MDA lifecycle begins from CIM and ends with code. It is depicted in Fig. 1.

MDA is a software development approach based on creation of models and transformations between them. Model transformation is one of the main issues and key elements in MDA.

CIM to PIM transformation is the first transformation in MDA and it is very important in designing high quality software. The Benefits of MDA appear in this transformation.

Several CIM to PIM transformation approaches have emerged, they have not been objectively analysed yet. This research presents an analytical review and evaluation of these approaches.

The main steps of the research are CIM to PIM transformation approach selection, development of evaluation criteria, and criteria-based evaluation of these approaches.

This paper is organized as follows: section 2 discusses existing CIM to PIM transformation approaches. Section 3 discusses a conceptual framework for CIM to PIM transformation. Evaluation of proposed approaches is discussed in section 4. Section 5 discusses conclusion of the paper.
2. Review of CIM to PIM transformation approaches

In this section, we discuss proposed approaches for transformation from CIM to PIM.

In [2] proposed a disciplined approach for transformation from CIM to PIM using feature-oriented and component-based approach. In this paper, feature model used for representing requirement in CIM and this model includes a set of features and relationship between them. Also software architecture used for representing PIM and it includes a set of components and interaction between them. Also responsibility considered as a connector between feature and component for facilitating transformation. It is depicted in Fig. 2.

![Figure 2. Proposed approach in [2].](image)

In [3] presented a possible solution for CIM modeling and then transform it to PIM using analytic method of transformation. In this paper, CIM level representing by DFD that it is used for business process modeling. PIM level represented by UML diagrams includes use case diagram, activity diagram, sequence diagram, and domain model. It is depicted in Fig. 3.

![Figure 3. Proposed approach in [3].](image)

In [4] proposed a disciplined approach for transformation of CIM into PIM. In this paper, CIM includes business process model and requirement model. First, business process modeled using an activity diagram then activity diagram details for specifying system requirement. PIM represented by class diagram. For this from the model of requirement elements the system components are created. Finally, a set of business archetypes help to transform the system components to the PIM layer in details. It is depicted in Fig. 4.

![Figure 4. Proposed approach in [4].](image)

In [5] presented an approach in which CIM is represented by secure business process in BPMN [6] and transform with the help of QVT [7] rules, checklists, and refinement rules into UML use case which represents PIM. It is depicted in Fig. 5.

![Figure 5. Proposed approach in [5].](image)

3. Conceptual framework of CIM to PIM transformation

We have used method of paper [8] for creating conceptual framework and extracting Evaluation Criteria from it.

We design a conceptual framework to extract Evaluation Criteria for comparing proposed approaches. The conceptual framework is composed of a "Static Model" describing common concepts and their relationships, "Taxonomies" of CIM and PIM for classifying, the types of CIM and PIM, and "Transformation Approach". The comparison and evaluation criteria are derived from this framework.

3.1 Static model

In this section, we formalize the concepts of CIM, PIM, transformation, and traceability by means of a meta-model. The meta-model is presented using the class diagram in Fig. 6.
As shown in Fig. 6, a CIM can be transformed into a PIM. A CIM and PIM are composed of one or more Model Element. Traceability Link is established between the model elements of the CIM and the PIM.

3.2 Taxonomies

The taxonomy of CIM and PIM classify different their aspects.

3.2.1 Taxonomy of CIM

In [9] is mentioned that Regarding CIM, there are two basic streams of suggestions of what is to be represented by CIM. One of the streams suggests that the business model is to be represented at this level [10]. Another stream points to CIM as a model, which represents system requirements [11]. Some researchers position both models representing business knowledge and system requirements at the CIM level [12].

In Fig. 7, we are depicted taxonomy of CIM.

As shown in Fig. 7, we classify the different representations of the Business Process Model aspect used in the proposed approaches into three types: UML [13] Diagram (Activity Diagram), Data Flow Diagram (DFD) and Business Process Modeling Notation (BPMN). The Requirement Model aspect is classified into two types: Use Case Model and Feature Model.

3.2.2 Taxonomy of PIM

A complete PIM should describe two aspects of a system: Structure and Behaviour. The structure (or static) aspect emphasizes the static structure of the system using classes, objects, attributes, operations, relationships, etc., while the behaviour (or dynamic) aspect emphasizes the dynamic behaviour of the system by showing interactions among objects, etc. As shown in Fig. 8, we classify the different presentations of the structure aspect used in the proposed approaches into three types: UML Structure Diagrams, Domain Diagram and Architecture Concept. The behaviour aspect is classified into one type: UML Behaviour Diagrams.

3.3 Transformation approach

CIM is further transformed into a PIM. As shown in Fig. 9 transformation in proposed approaches is classified into Rule-Based transformation. Rule-Based transformation uses a set of predefined Transformation Rules.

A transformation rule is a description of how one or more constructs in the source language can be transformed into one or more constructs in the target language [14].

4. Criteria-Based evaluation of CIM to PIM transformation approaches

In this section, first we derive evaluation criteria from conceptual framework in section 3 and then compare proposed approaches.

4.1 Evaluation criteria

The evaluation criteria are used to evaluate each proposed approach in terms of Coverage of CIM, Completeness of PIM, Automation, Evaluation Methods, Traceability Support, and...
Tool Support. As shown in Fig. 10, for example, the evaluation criterion “Completeness of PIM” is derived from the Taxonomy of PIM. Notice that the Static Model of the conceptual framework is not directly related to any evaluation criteria; however, the taxonomies are all dependent on it. Last, note that the criterion “Evaluation Methods” in proposed approaches reports, and it is not therefore traced back to the conceptual framework.

4.1.1 Evaluation Criterion for CIM

We evaluate the CIM with respect to “Coverage of CIM”. The evaluation criterion “Coverage of CIM” is derived from the Taxonomy of CIM. “Coverage of CIM” indicates which aspects of CIM covers by proposed approaches.

4.1.2 Evaluation Criterion for PIM

We need to evaluate the generated PIM with respect to its completeness. If a generated PIM both describes the system structure (e.g., class diagram) and behaviour (e.g., sequence diagram, state machines, or activity diagrams), then we label the generated PIM as complete. If a generated PIM describes only one of these two aspects of a system (i.e., either the structure or behaviour), then we label it as incomplete.

4.1.3 Evaluation Criterion for Transformation

We evaluate the transformation approaches proposed in the approaches with respect to their automation. The automation criterion evaluates whether a transformation is automatic, semi-automatic, or manual. A transformation approach is automatic if it has been fully implemented. A transformation is semi-automatic if there is algorithm and transformation rule for proposed approach and user interventions are required. Last, some approaches are entirely manual.

4.1.4 Other Evaluation Criteria

We evaluate the approaches with respect to “Evaluation Methods”, “Traceability Support”, and “Tool Support”.

If a paper applies other evaluation methods (besides case studies) to evaluate its approach, we mention it in “Evaluation Method” criterion.

4.2 Discussion

In Table 1 we summarize the comparison of Evaluation Criteria for proposed approaches in tabular form. In this subsection we describe results from Table 1.

- **CIM**
  
  We can see from Table 1, Two out of four approaches can cover business model (e.g., DFD). One approach can cover requirement model (e.g., Feature model). One approach is capable of covering CIM, including both aspects.

- **PIM**
  
  We can see from Table 1, that two out of four approaches can derive only structural model elements (e.g., objects, classes, associations, components) from CIM. One approach can generate behavioural features of a system (e.g., sequence diagrams, state machines, and/or activity diagrams). One approach is capable of generating PIM including both structural and behavioural aspects of a system, which is characterized as complete according to our evaluation criteria;

- **Transformation–automation**

  As shown in Table 1, two out of four approaches require user intervention to semi-automatically perform the transformation; two approaches require manual transformations.

- **Evaluation Methods**

  All of the approaches have their transformation approaches evaluated. Case study has been performed to evaluate all the approaches.

- **Traceability support**

  All of the approaches do not propose any method for traceability support.

- **Tool support**

  All of the approaches do not develop any tool for supporting their approaches.

5. Conclusion

In this paper, we have surveyed several CIM to PIM transformation approaches and have evaluated them using a set of evaluation criteria. For this, we have designed a conceptual framework and have derived Evaluation Criteria for comparing proposed approaches. The results can be used for
assessing and comparing CIM to PIM transformation approaches.

In the future work we want to propose a semi-automatic approach for CIM to PIM transformation that generates a complete PIM.

References

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Business Model</td>
<td>Requirement Model</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Wei et al. [2]</td>
<td>Feature Model</td>
<td>Yes</td>
<td>Software Architecture</td>
<td>Yes</td>
<td>Semi-automatic</td>
<td>No</td>
<td>Case Study</td>
<td>No</td>
</tr>
<tr>
<td>Kardoš et al. [3]</td>
<td>DFD</td>
<td>Yes</td>
<td>Use Case Diagram Sequence Diagram Activity Diagram Domain Model</td>
<td>Yes</td>
<td>Yes</td>
<td>manual</td>
<td>No</td>
<td>Case Study</td>
</tr>
<tr>
<td>Kherraf et al. [4]</td>
<td>Activity Diagram Use Case Diagram</td>
<td>Yes</td>
<td>Yes</td>
<td>Class Diagram</td>
<td>Yes</td>
<td>manual</td>
<td>No</td>
<td>Case Study</td>
</tr>
<tr>
<td>Rodríguez et al. [5]</td>
<td>BPMN</td>
<td>Yes</td>
<td>Use Case Diagram</td>
<td>Yes</td>
<td>Semi-automatic</td>
<td>No</td>
<td>Case Study</td>
<td>No</td>
</tr>
</tbody>
</table>

1 Representation