Refactoring is important to software reuse and software reuse is important to refactoring. Previous works on refactoring definitions have not been excluding.

Automated refactoring tools are software system that works with a particular family language. They define refactorings and support for reuse based on frameworks. The architecture has been changed since initial work due to an iterative process.

Related works on refactoring are:
- Refactoring definitions are non-formal or semiformal (pre- and postconditions are expressed as conditions and functions on model language primitives defined on MOON grammar).
- Works with a particular language such as Java, Smalltalk.
- Do not reuse refactoring definitions.

**Problem:** Loss reuse from previous efforts on refactoring definitions when adapting to new source language.

**MOON Refactoring Framework**

**MOON Core**

**Java Extension**

**Eiffel Extension**

**Engine Core**

**Refactoring Repository**

**MOON Refactoring Framework**

**previous works**
- Model language MOON
  - Abstract constructions in the refactoring definitions
  - Common to a family programing languages
  - Refactoring parameterization [Crespo, 2000]

**Related works**
- Refactoring definition
  - Non formal
  - Semiformal (pre- and postconditions are expressed as conditions and functions on model language primitives defined on MOON grammar).
  - Works with a particular family language.
  - Tools for Java, Smalltalk.
  - Do not reuse refactoring definitions.

**Example:** Specialize Bound S

**Model Language: Template**

**Description**

Motivation

**Inputs:** C, G, B

**Preconditions:**

\[ G \in \text{FormalParam}(C) \]

\[ \forall T \in \text{SubstFormalPar}(C, G) \Rightarrow T \in \text{Subtype}(B) \]

\[ \text{Postconditions:} \]

ReplaceBoundType(C,G,B)

**Example:** Predicates and functions

**Example:** Actions

**Example:** MOON Framework and extension with...