Tool Support for SDL Patterns

D. Cisowski, B. Geppert, F. Rößler, M. Schwaiger

Contents:

- Potential for tool support in general
- Pattern application - an example
- SPEEDI - SDL pattern editor
- Conclusion

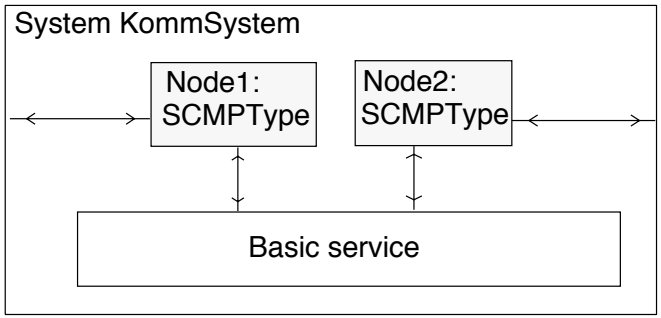
Potential for Tool Support in general

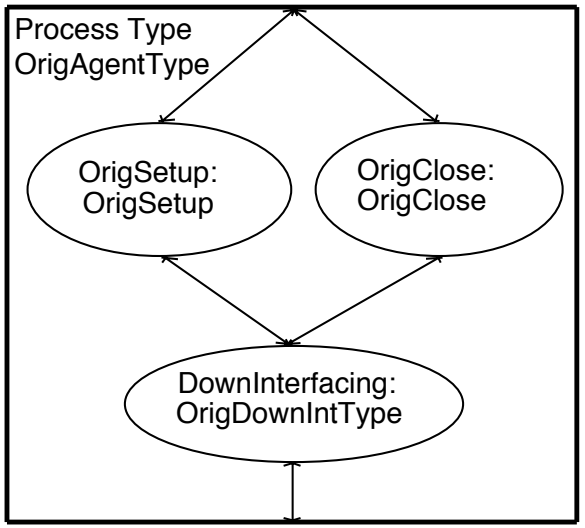
Possible features:

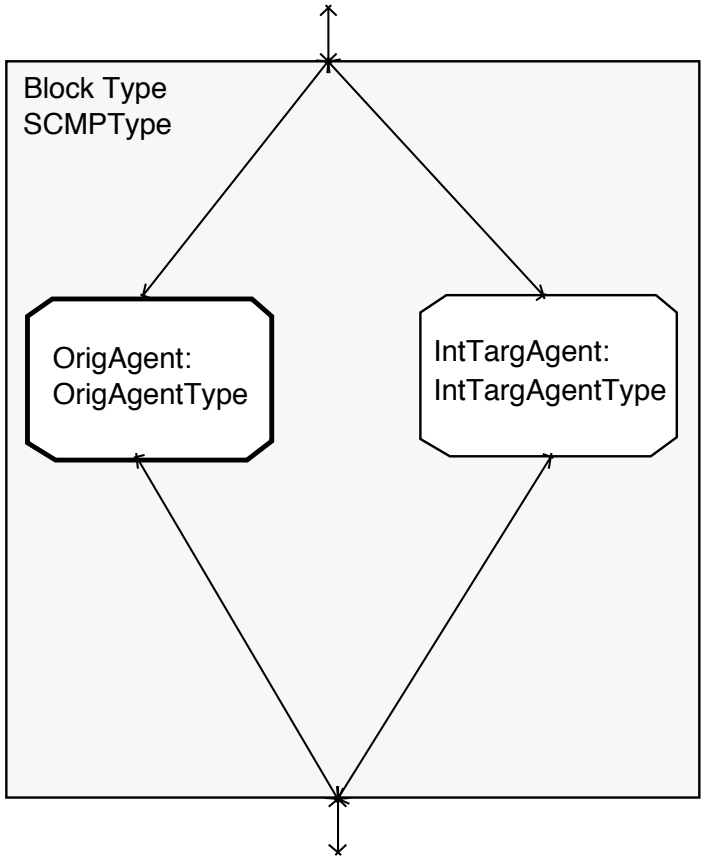
- Pattern development
- Pattern documentation
- Pattern application
  - Pattern selection
  - Pattern adaptation and composition
  - Pattern validation
- Code generation
- Quality improvement of SDL patterns and configuration process

Pattern Application

- ST2+ Step VII: Context Specification & Design Problem -



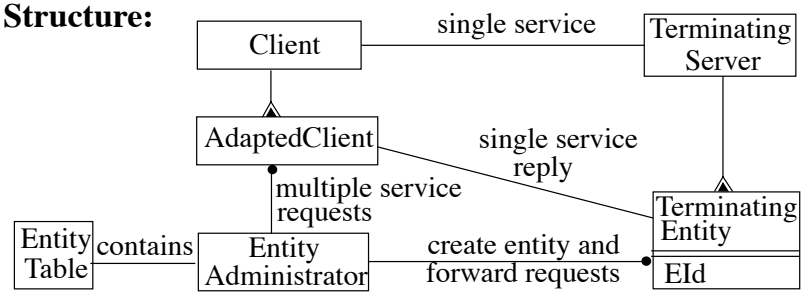




Pattern Application
- ST2+ Step VII: Selected SDL Pattern (extract) -

Name: DynamicEntitySet

Structure:



Syntactical embedding:

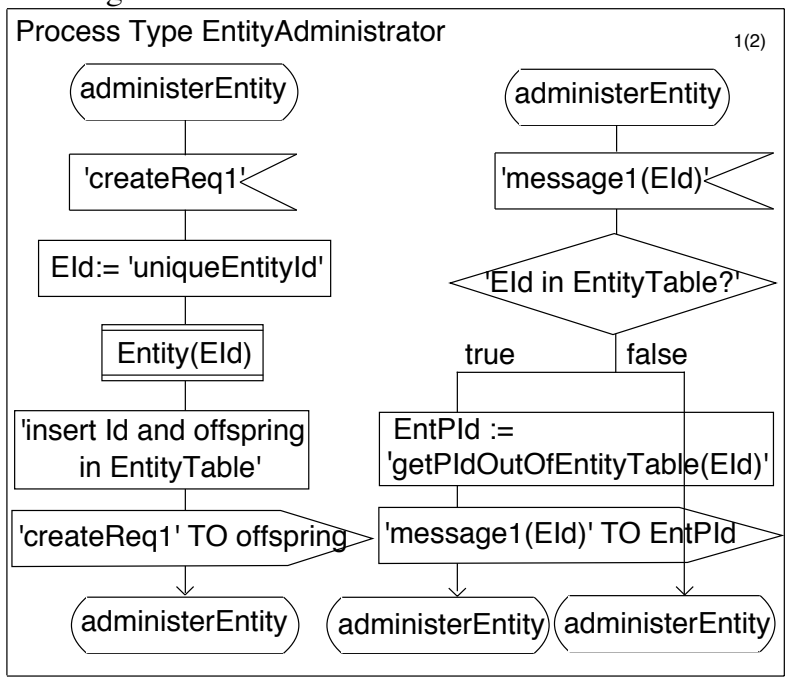
Specialization: transitions of *TerminatingServer* which send a signal back to the *client* are potential candidates for redefinition in order to inform the *client* about the local *EId*. The protocol engineer has to decide which ones are relevant or if the client is informed otherwise. In any case the *EId* will be used by the *AdaptedClient* when sending signals to the *TerminatingEntity*. Therefore all transitions which send a signal (except *createReq1*) to *TerminatingEntity* are redefined by adding the *EId* as signal parameter.

A process of type *EntityAdministrator* is added to the surrounding block diagram of *TerminatingServer*.

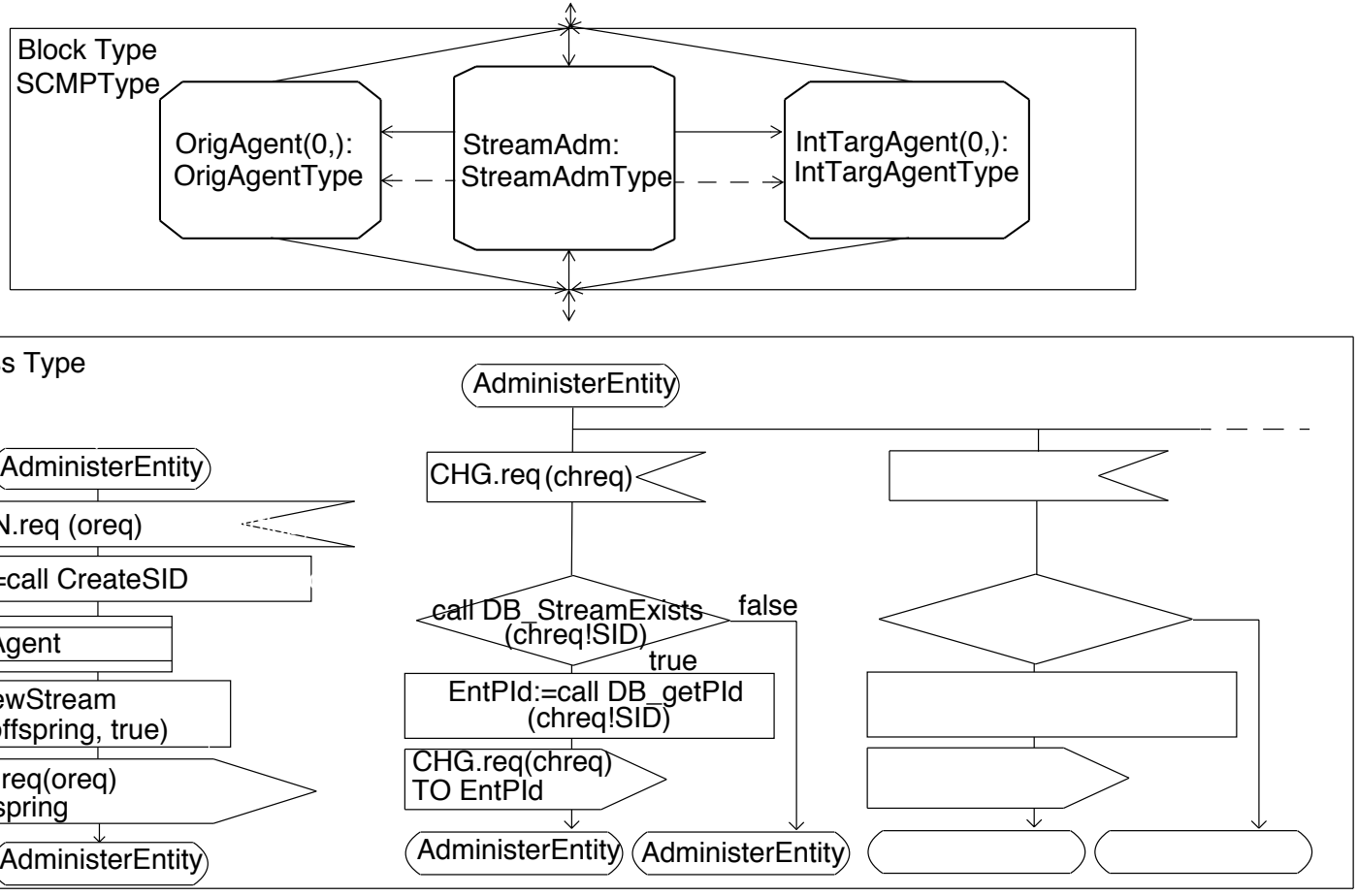
Renaming: *createReq1* and *message1* correspond with those messages the *client* sends to its *TerminatingServer*, where *createReq1* is the first message received. However, the concrete quantities of course have to be adapted.

Structural change: signal routes to *TerminatingServer* must be deleted and redirected to *EntityAdministrator*. The reference symbol for *TerminatingServer* must be replaced by a process set reference *Entity* with corresponding process type *TerminatingEntity* in the embedding block. *EntityAdministrator* must be connected with the process set *Entity* by a create line and additional signal routes for forwarding the messages.

SDL-fragment:

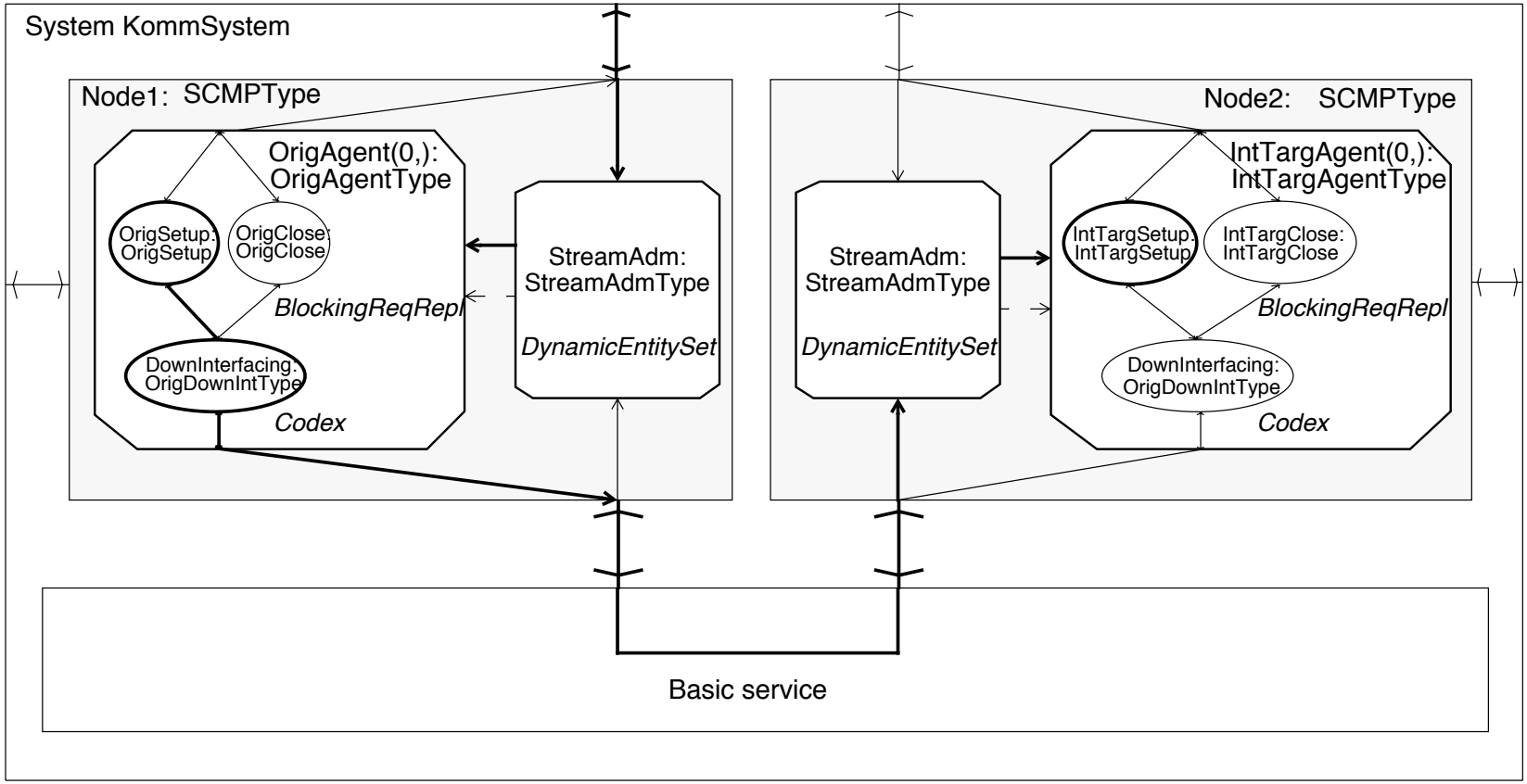


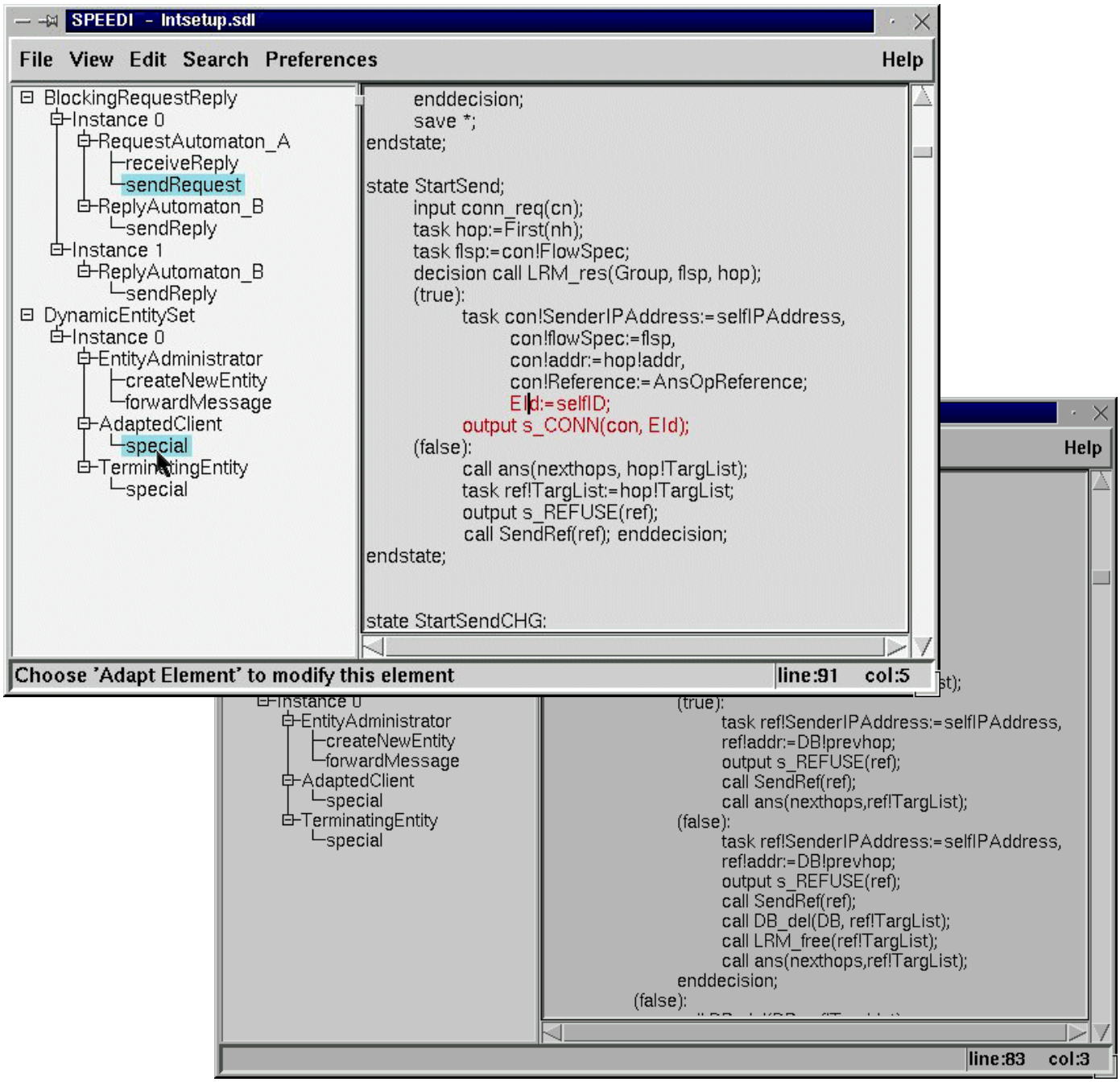
Pattern Application
- ST2+ Step VII: Adaptation and Composition -

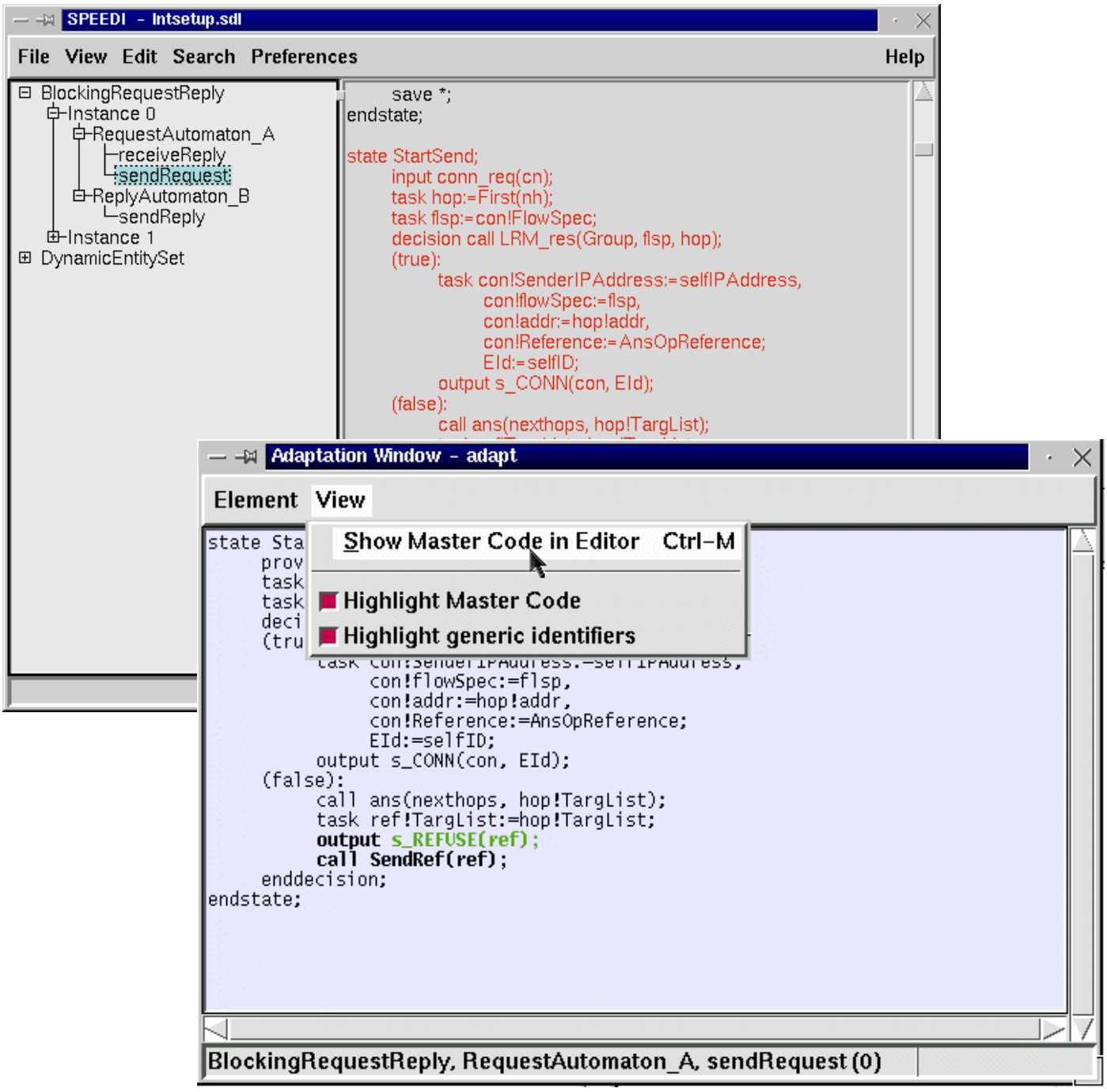


Pattern Application

- ST2+ Step VII: Pattern Documentation-







SPEEDI - SDL Pattern Editor

- Features -

Pattern documentation

- Tree-like view of
  - the specification's syntactical units with pattern instances
  - used patterns
- Collapsing and expanding with jump functionalities
- Coloring
- Clustering

Pattern adaptation and composition

- List of context identifiers
- Observation of syntactical embedding and refinement rules

Conclusion

Design decisions

- Supporting of SDL/PR
- Integration into an existing SDL tool environment with a suitable parser & semantic analyser

Implementation issues:

- Programming language - Python
- Library for building graphical user interfaces - Tcl/Tk (Tkinter)
- Portability

Current development state

- Pattern documentation