# ASAP Online Toolkit

EU Future & Emerging Technologies Project IST-2001-38059

## Overview

In the FET funded project ASAP we have developed a generic toolkit and made it available online. An integrated web interface, comprising the individual tools described below, is accessible at: http://clip.dia.fi.upm.es/Projects/ASAP/.

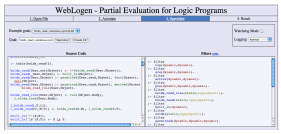
## CiaoPP

CiaoPP is the abstract interpretation-based preprocessor of the Ciao multi-paradigm program development environment. CiaoPP can perform a number of program debugging, analysis, and source-to-source transformation tasks on (Ciao) Prolog programs.

## Ecce

ECCE is an automatic online program specializer for pure Prolog programs (with built-ins). It takes a pure Prolog program and a query of interest and then specializes the program for that particular query. It can also be used to slice a program for a particular query. Ecce can perform deforestation and tupling and uses homeomorphic embedding and characteristic trees for automatic control.

## Logen




This is an offline specializer for almost full Prolog. It works on an annotated version of the source program and the web interface allows user-friendly editing of the annotations. Logen is a very efficient system, producing specialized specializers that can be compiled and distributed independently of Logen itself.

## PolyTypes

The purpose of the PolyTypes tool is to compute a well-typing for a given module, namely a set of type rules together with a type signature for each program predicate. A well-typing is interpreted in the usual sense of Hindley-Milner types. The types generated by PolyTypes are for example accepted by the type checker for Mercury, a typed logic programming language.

## NFTA

The NFTA (Nondeterministic Finite Tree Automata) tool computes a regular approximation of a given program and uses this to detect useless clauses. The tool computes descriptions of the sets of terms that can occur in the heads of clauses in computations of the program, as non-deterministic regular tree grammars (or NFTAs). These are both more complex and more accurate than classical regular approximations which compute (top-down) deterministic regular tree approximations.

## Domain Model

The purpose of the Domain Model tool is to compute an abstract model of a program over a set of finite symbolic values such as modes or simple types. The abstraction is determined by an arbitrary set of regular type rules.