

# Esterel

Robert de Simone  
INRIA Sophia-Antipolis

## 1. History

Esterel is an imperative *synchronous* language, meant to specify and program real-time reactive systems. It was invented around 1981 in the *Centre for Applied Mathematics* of the *Ecole Supérieures des Mines de Paris*, located in Sophia-Antipolis, in southern France, near the Mediterranean Esterel hills chain. Its fathers were mainly Gérard Berry, Jean-Paul Rigault, and Jean-Paul Marmorat, with further team members soon joining.

The language was further developed during the 1980s at Ecole des Mines and INRIA, together with its formal semantics owing to the Structural Operational Semantic (SOS) definitional style of process algebras. It was then mainly applied for embedded monitoring systems, at the crossroads of computer science and control theory. In the early 1990s the relevance of the synchronous programming paradigm for hardware design and synthesis was further recognized, and progressively more and more methods from each side were applied to the other (incidentally, the Verilog and VHDL hardware description languages are Esterel contemporaries, but with a less strict synchronous approach).

A graphical form of the language with a StateChart look-alike flavour, but a strict and sound synchronous semantics, thereby named SyncCharts, was developed by Charles André at the nearby University of Nice around 1996. Early adoption by a consistent set of industrial partners led to the creation of a spin-off software editing company under the name of Esterel Technologies, founded in 1999. Later this company acquired the SCADE compiler and its development team. SCADE is the industrial version of the Lustre language, another synchronous reactive language, with a declarative/block-diagram syntax complementary of the Esterel imperative (C-like) style. Recently a specific subsidiary company under the name of Esterel-EDA was created to address the hardware ESL EDA market.

## 2. Description

The following example is classically used as an introduction to Esterel programming style:

```
Module ABRO;  
input A, B, R;  
output O;  
  
loop  
  abort  
    ( await A || await B );  
    emit O  
  when R  
end.
```

The example shows a number of things: first, there is an explicit “parallel” (||) programming construct; then there are specific signal types, and signals can be emitted or received to progress control beyond some points. The main paradigm beyond the synchronous languages philosophy is

that time is divided into discrete (logical) *instants*, that these instants are shared between parallel components, and that the joint behaviour created during an instant is called a *reaction*. The issue of simultaneity between several events (signals occurring during the same reaction), and proper handling of priority and causality between such simultaneous events is fully considered in the semantics. Here, the output signal **O** will be emitted each time signals **A** and **B** have been notified, simultaneously or one after the other, from the environment. Actually this is only so unless the reset signal **R** occurs, which erases all effects from previous receptions of **A** or **B**, *including the ones at the current instant*.

The instruction **await Sig** can further be expanded as **abort loop pause when Sig**. This exposes the specific **pause** statement, which is the *single* primitive statement which allows dividing the progress of executions over time into successive instants. Then any emitted signal, be it sensed from the external environment or explicitly emitted from a parallel branch inside, is considered as present for a lifetime of that instant (only). Signal presence is broadcast, and thus perceived uniformly all across the program. Parallel subprograms may proceed with their activities while put under the guard of a possible interrupting signal (as in **abort P when S**). The fine tuning and precise handling of priorities is allowed by the primitive constructs (as in **weak abort P when S** vs **abort P when S**, the former pre-empting its body only at the end of the reaction, while the latter discards it right away for the reaction when **S** occurs).

Various primitive constructs allow to start, freeze, abort or reset a subprogram as needed by the problem specified. Esterel is based on a clear notion of instantaneous causality, which allows to discard some programs as ill-formed. The prototypal ill constructs are **present S then emit S** and **present S else emit S**, where forward executions is needed to validate or invalidate a branching decision based on a signal.

Data handling in Esterel is deferred to C/C++ as host languages. Even though the presentation was historically drastically different, this is similar to SystemC, where several primitive constructs and a scheduling/simulation/run-time mode are appended to construct parallel, concurrent, and synchronous tasks communicating as to form the global program/system behavior.

### 3. Conclusions and Future

The language is currently being standardized at IEEE, under the [P1778 Esterel standardization project](#). Recent extensions have been proposed to encompass multiclock design, and to combine the imperative features of Esterel with the declarative ones of Lustre in a fully integrated framework.

The classical compiling schemes into Mealy FSM or Boolean equation code, both fit for either software or hardware design, are being complemented with a Fast C compiler. The new compilation scheme aims at obtaining improvements of the same orders of magnitude in simulation speed as high-level TLM SystemC versus low-level Verilog/VHDL, but while retaining a logical cycle-based execution framework.

### 4. Links and References

The commercial compiler is distributed by the Esterel-EDA company. Further documentation and announcements can be found at <http://www.esterel-eda.com/>

Apart from the many academic articles and presentations, the definite reference on Esterel is currently: [Compiling Esterel](#), by D. Potop, St. Edwards, and G. Berry.

The book (in French) [Programmation synchrone de systèmes réactifs avec Esterel et les SyncCharts](#), by L. Zaffalon, is a programming manual with examples from the robotics area.