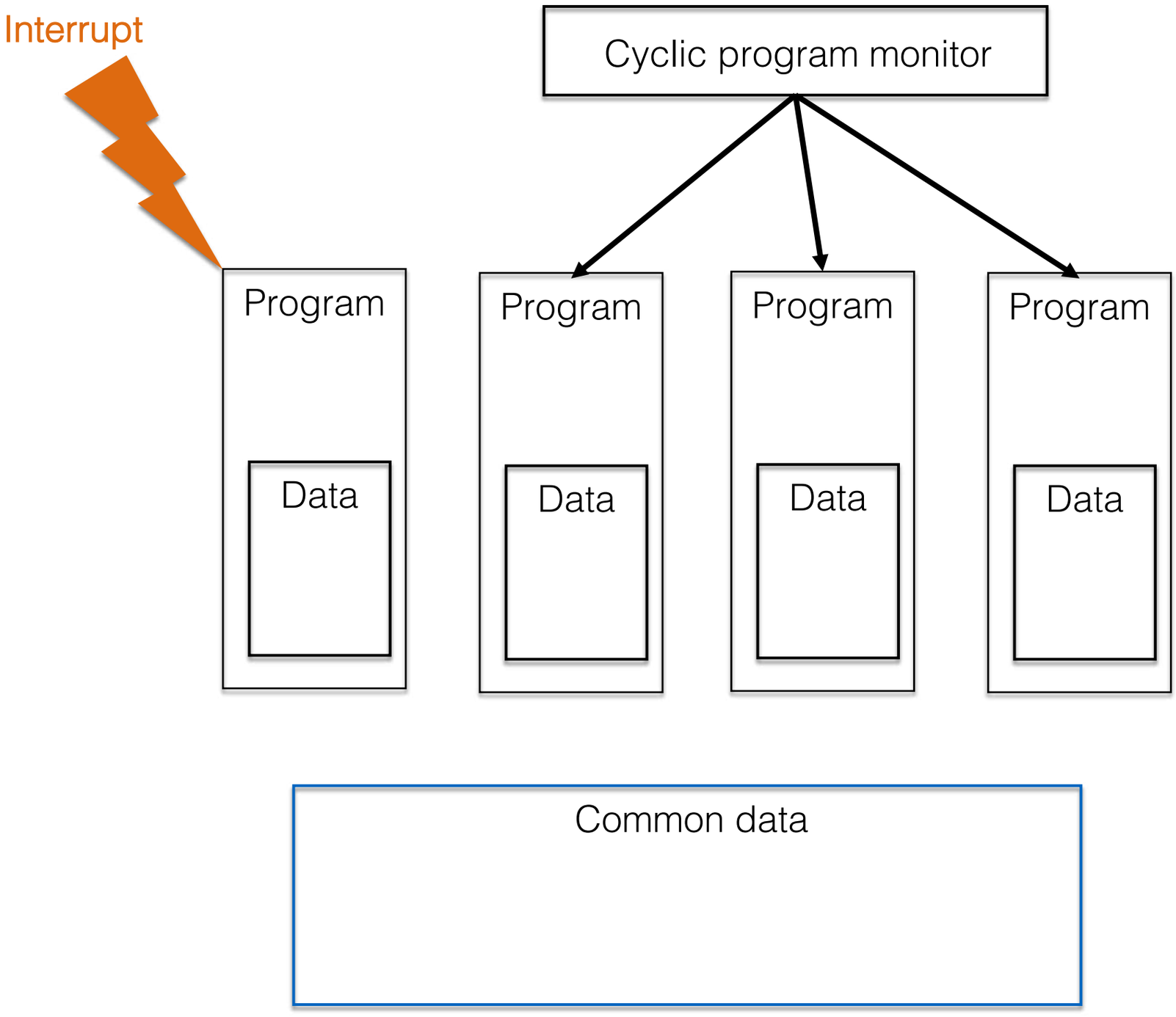
Concurrency before Erlang

or

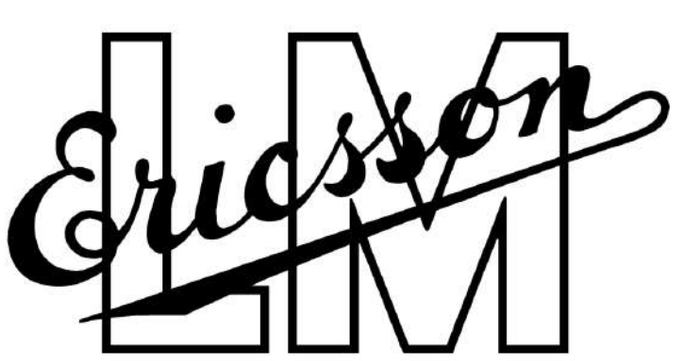
How Erlang got its name

Bjarne Däcker
former manager of CSLab

Program structure of a simple control system



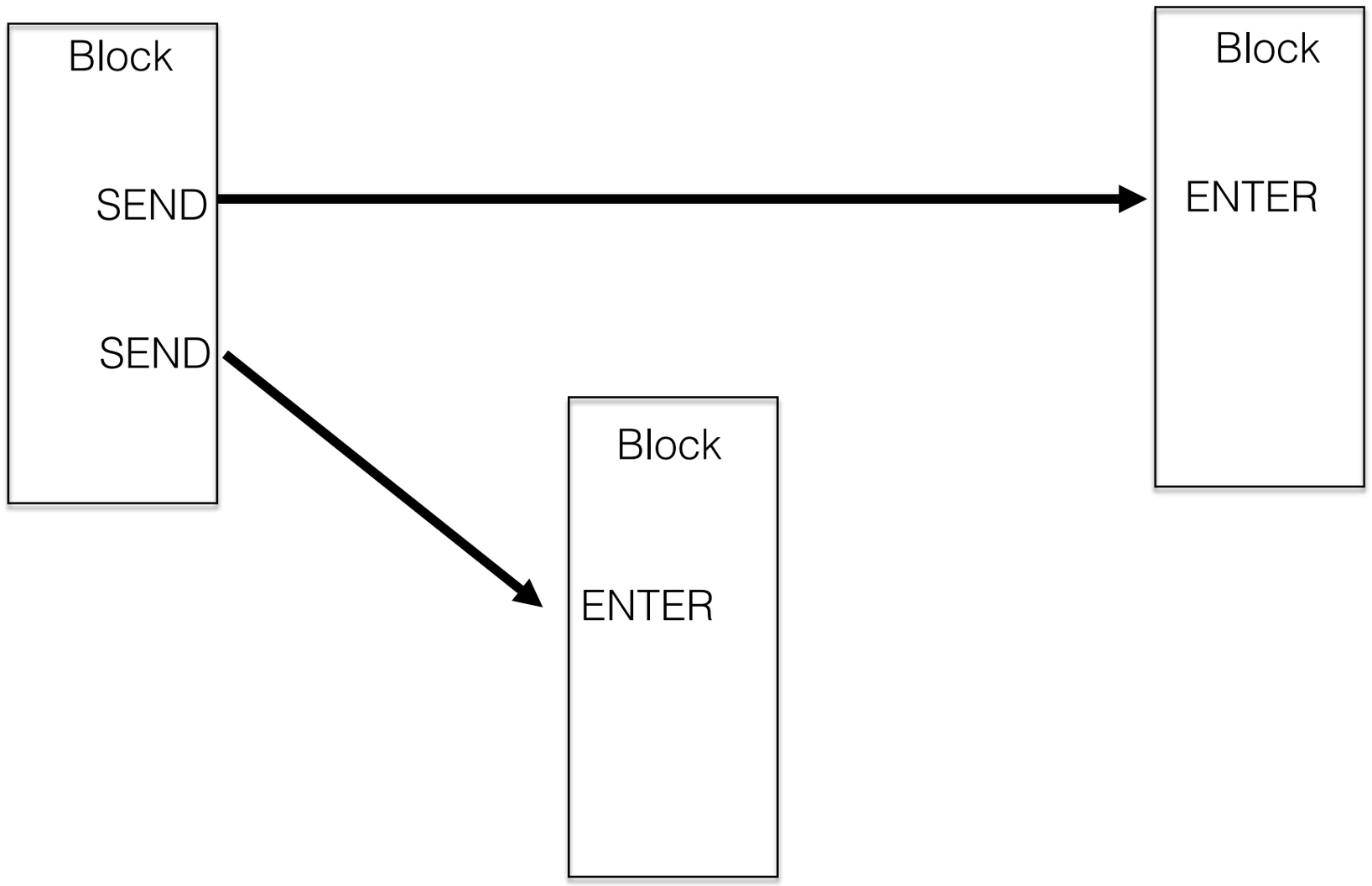
Each program has one entry point and has to keep its own state information



PLEX

Released early 1970's

Used for the AXE10 system



A block was like a combined module and static process.

Like cooperating finite automatons.

Professor of
Computer Science at
ETH in Zürich.

Inventor of Pascal
and other languages.

What we learnt from Niklaus Wirth and Modula

Modules

Processes

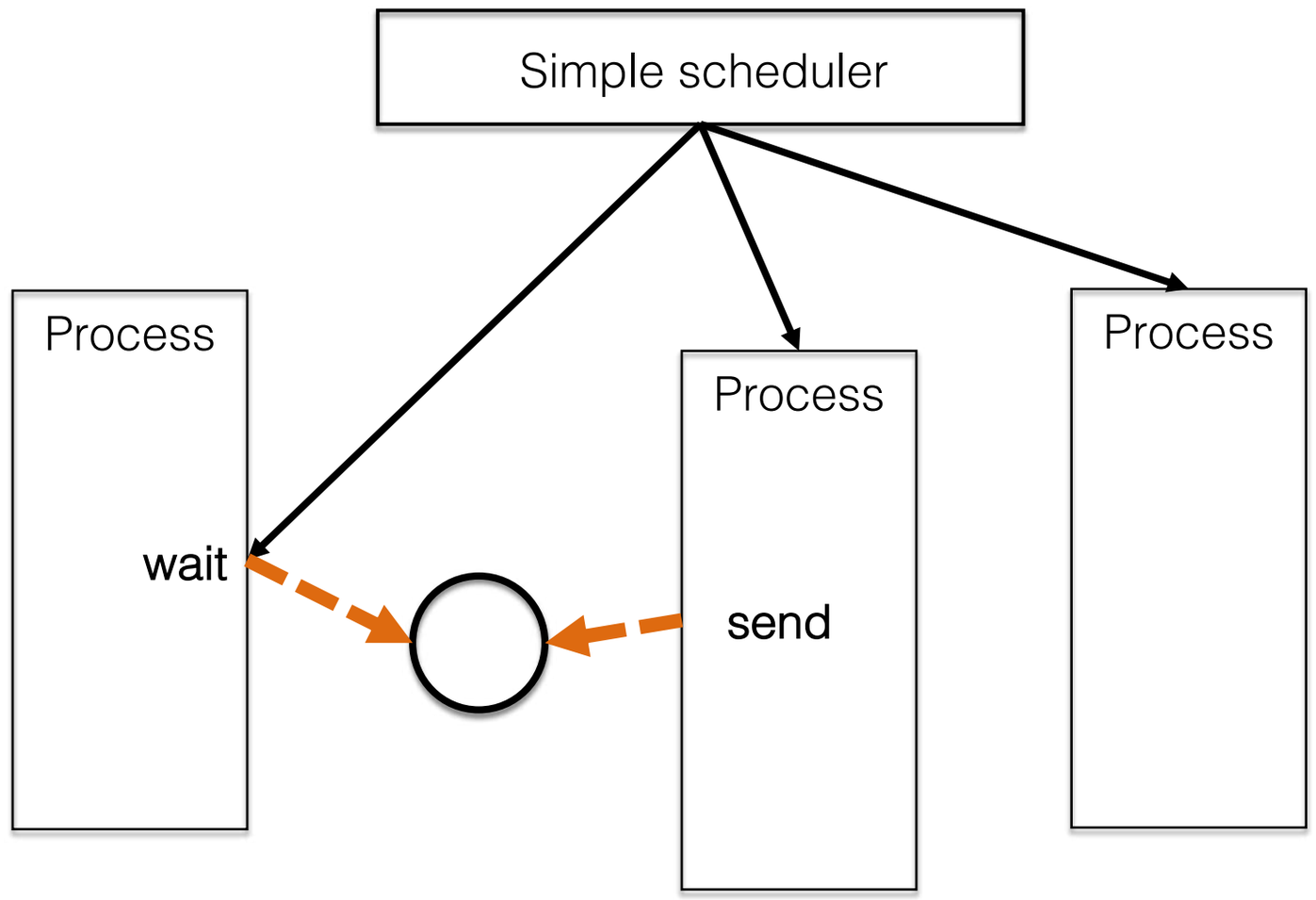
Process Communication

PL-type languages

PL163
Released early 1970's

Processes

Buffers of signals



High-level assembler

A process can be suspended waiting for a signal. When that arrives the processing continues

Modula

Invented by Niklaus Wirth
Released early 1970's

Modules

Processes

Process Communication
through interface modules
and signals

Interface module

procedure

procedure

Only one procedure can be executing. Others will be suspended

Presenting PASTEL (*Pascal for Telecom*) for Niklaus Wirth in May 1976

CHILL

Developed for CCITT
Released late 1970's

Modules

Processes

Process Communication
Three methods

- Regions
- Buffers
- Signals

There are several reasons why CHILL provides three different mechanisms for process communication:

- The ideas about what is the best method of communication between processes have not yet been stabilized in the world of programming language design. It would be too early to supply only one method of communication.

- Experience with communication between processes in a distributed system (without common memory between processors) is very limited. One communication mechanism may not be able to function optimally in both distributed and common memory architectures.

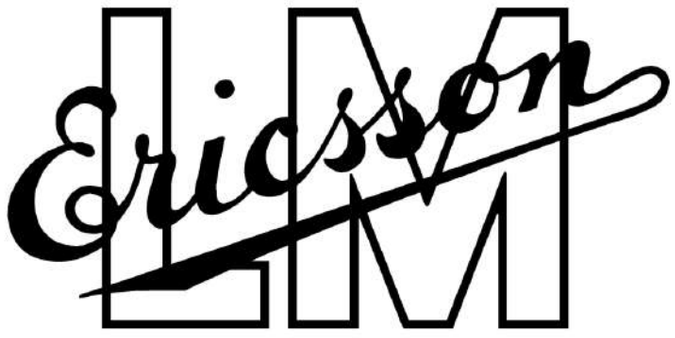
There are several reasons why CHILL provides three different mechanisms for process communication:

- The ideas about what is the best method of communication between processes have not yet been stabilized in the world of programming language design. It would be too early to supply only one method of communication.

- Experience with communication between processes in a distributed system (without common memory between processors) is very limited. One communication mechanism may not be able to function optimally in both distributed and common memory architectures.

In fact,

- Regions (Philips)
- Buffers (ITT)
- Signals (Ericsson)



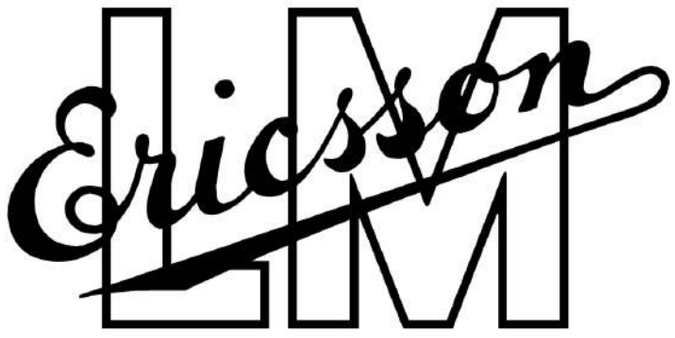
EriPascal

Released late 1970's

Modules

Processes

Process Communication
using signals



EriPascal

Released late 1970's

Modules

Processes

Process Communication
using signals

Semantically equivalent to a subset of CHILL
Implementation based on USD Pascal
and a runtime system called EriOS

Ada

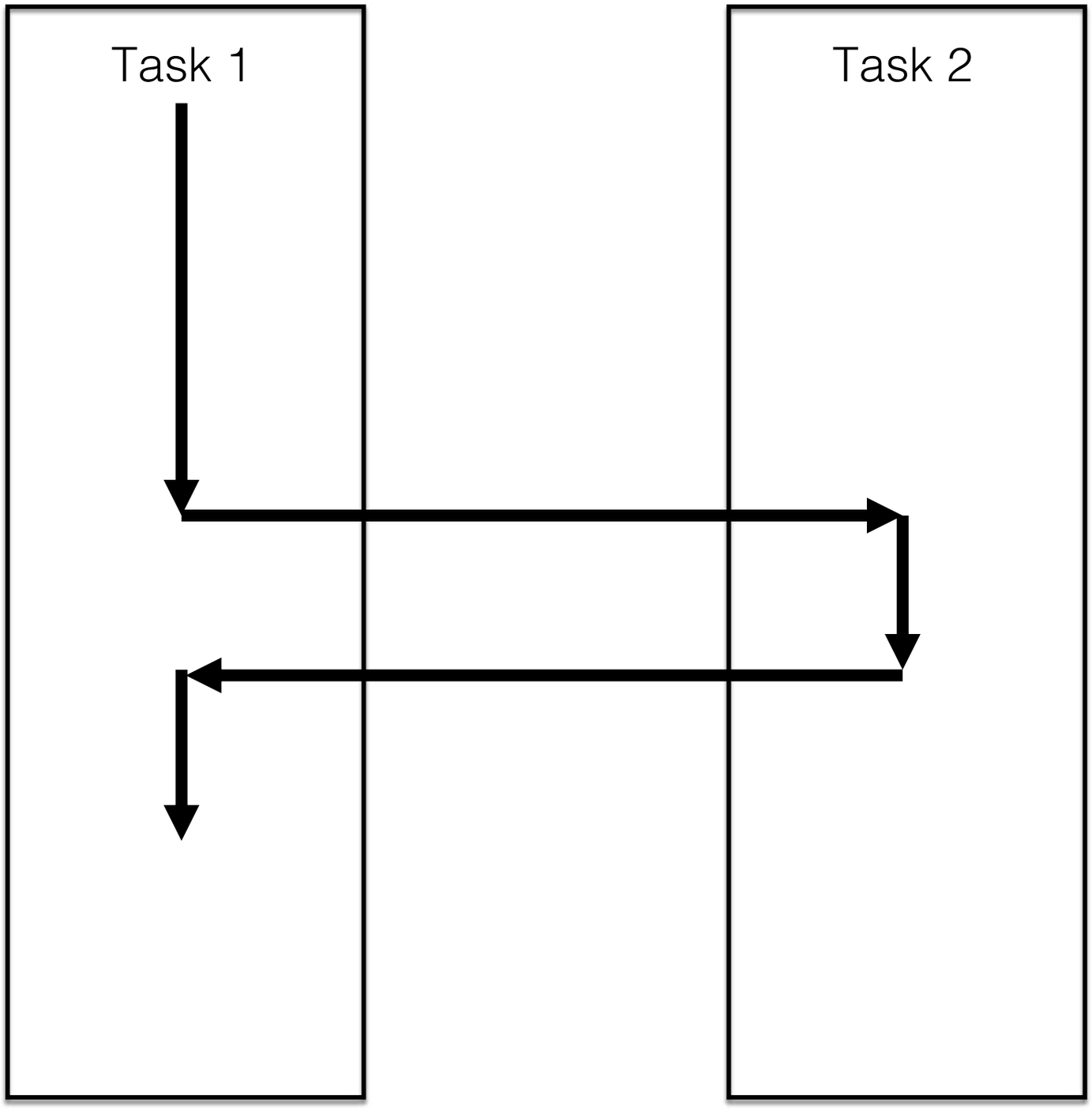
Developed for US DoD
Released early 1980's

Packages

Tasks

Task communication
through *rendez-vous*

Rendez-vous in action



Task 1 makes a procedure call to Task 2. The two tasks synchronize at this point.

Implementing message passing using *rendez-vous*

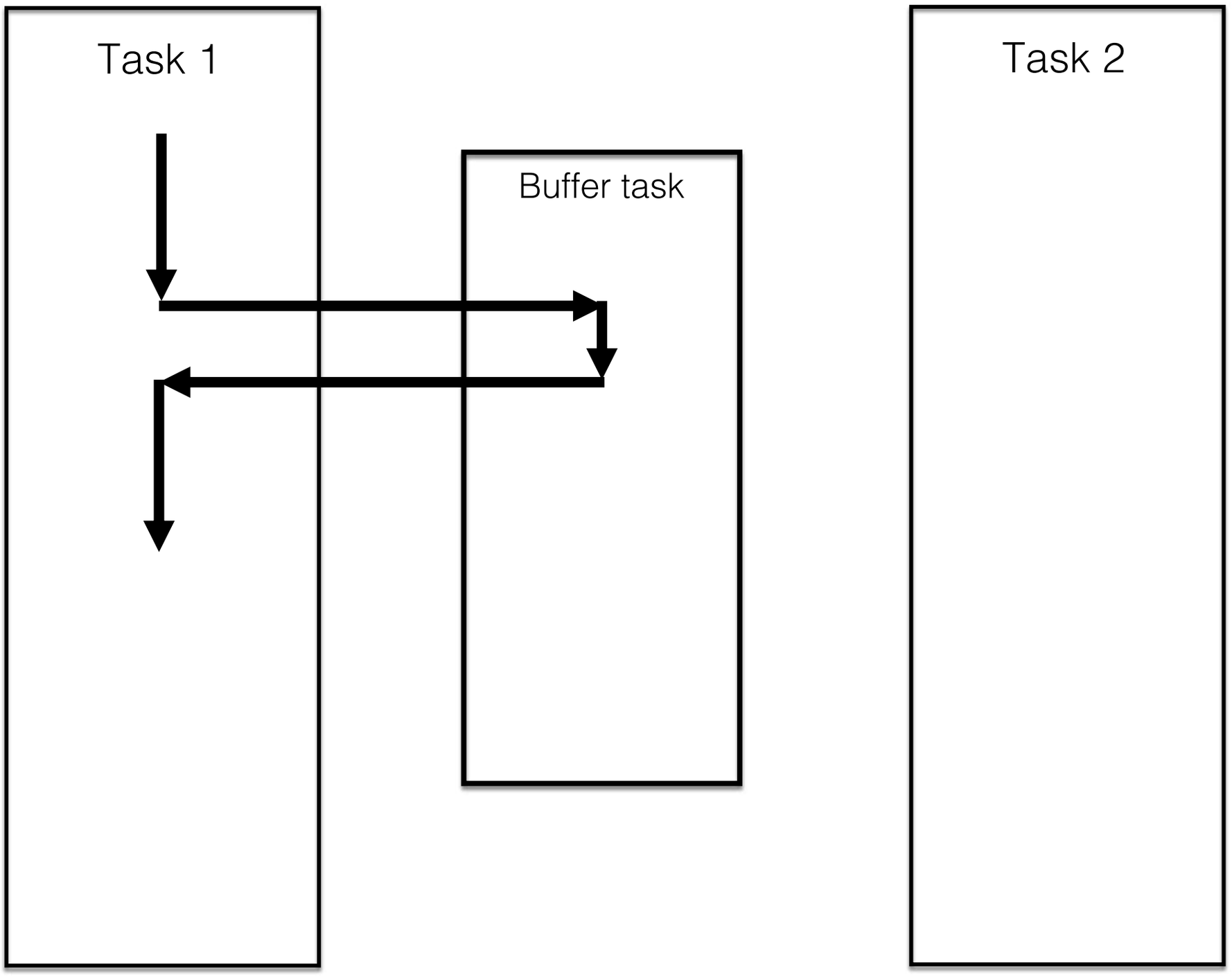
Task 1

Buffer task

Task 2

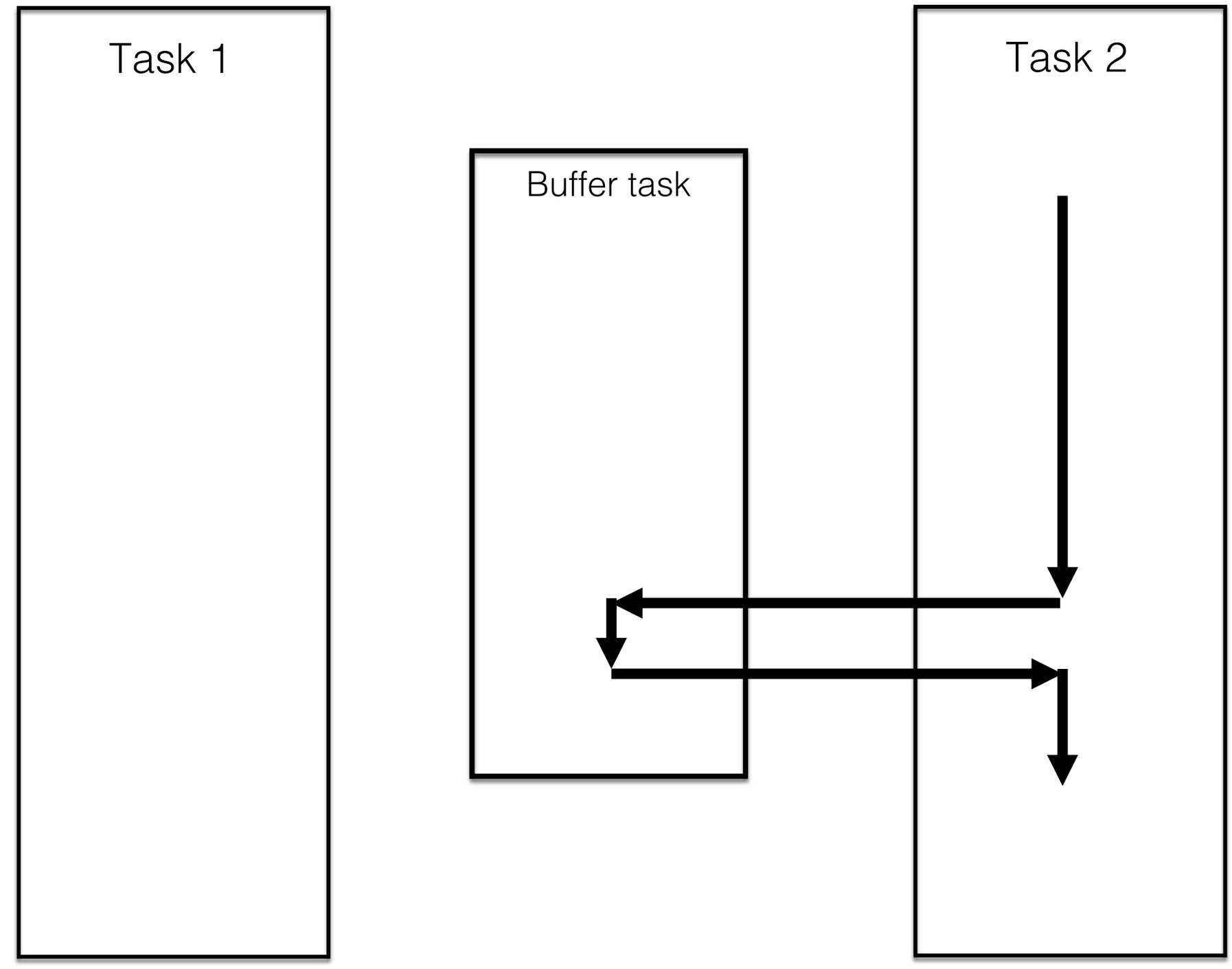
Message passing requires a buffer task

Implementing message passing using *rendez-vous*



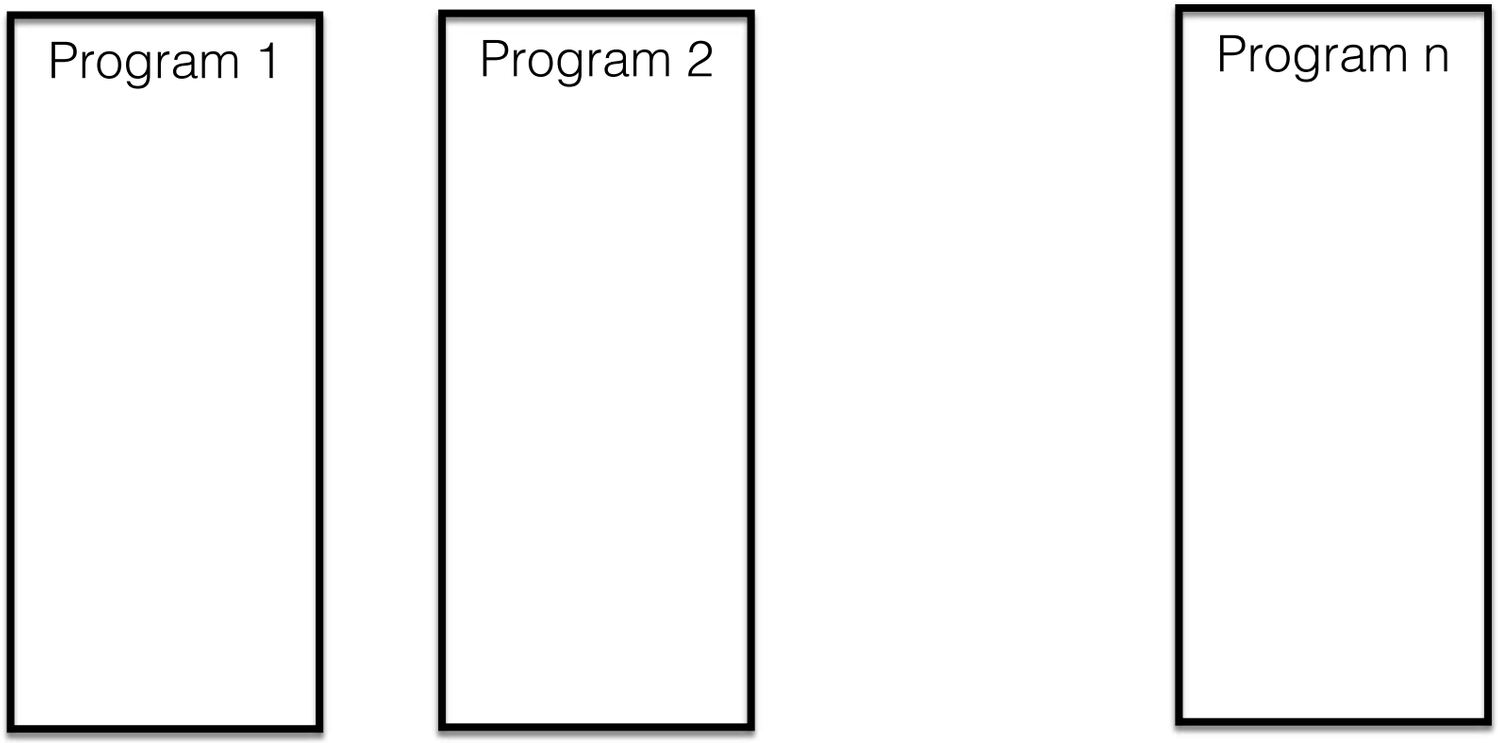
Message passing requires a buffer task

Implementing message passing using *rendez-vous*

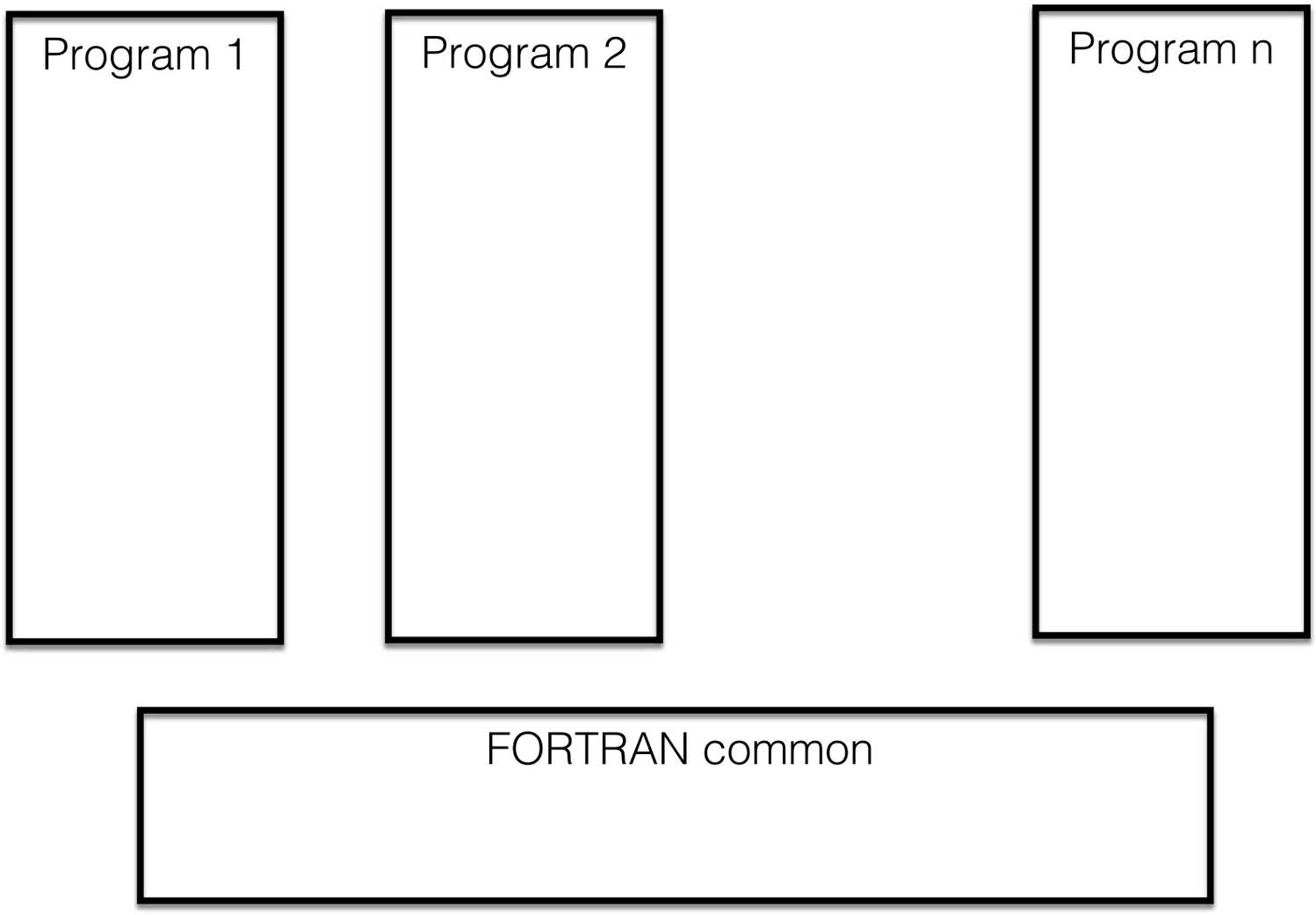


Message passing requires a buffer task

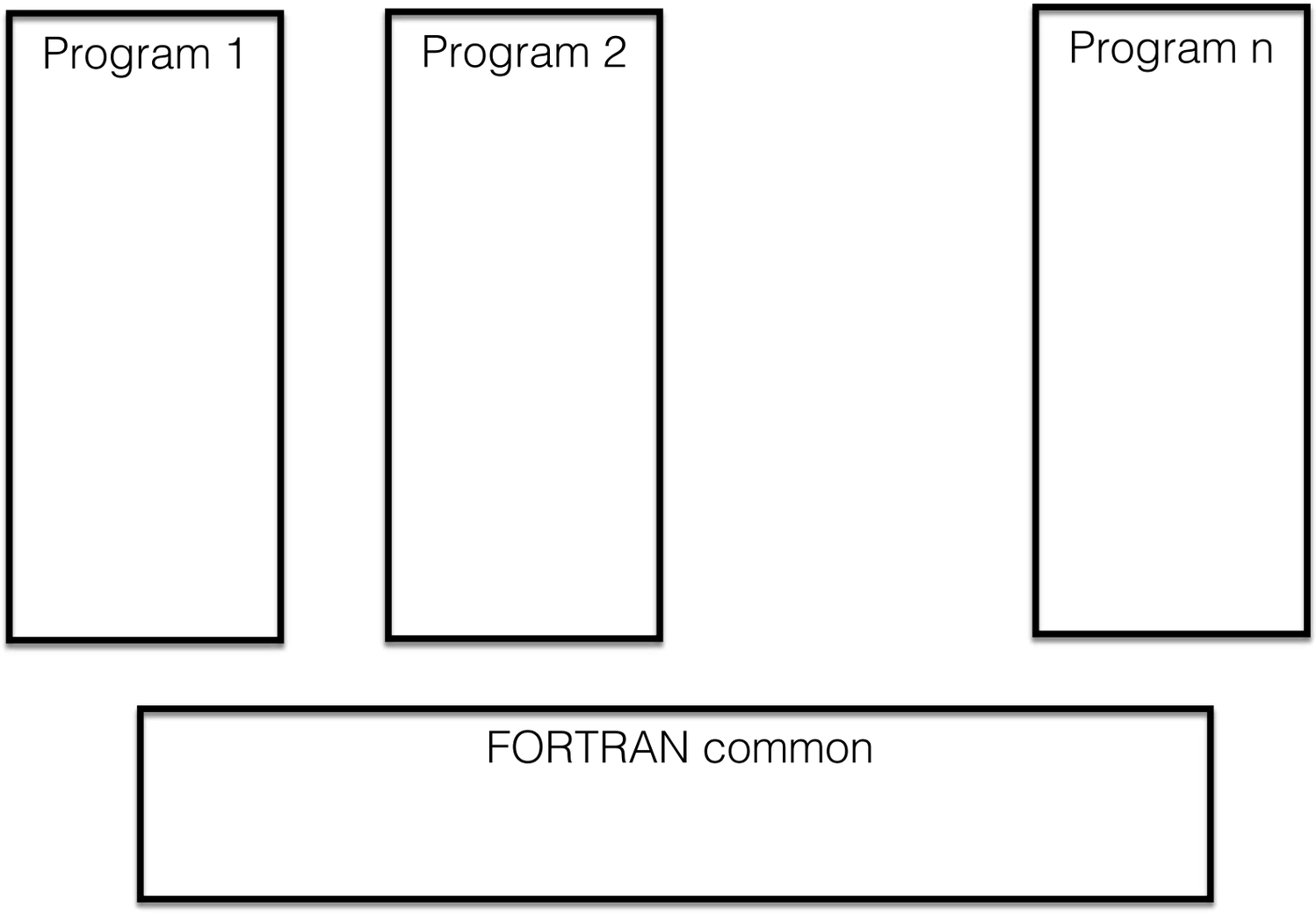
- Why was process communication so complicated?
- Because most Computer Scientists were thinking in terms of sequential programming.



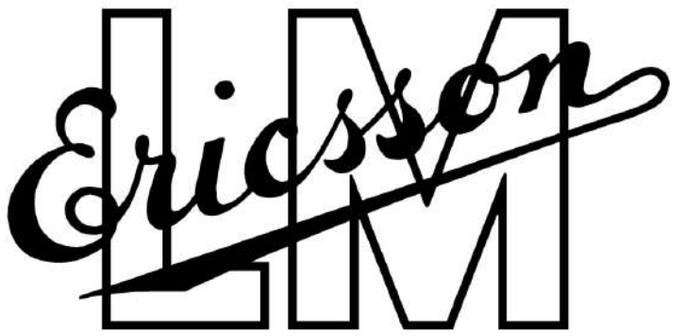
- Why was process communication so complicated?
- Because most Computer Scientists were thinking in terms of sequential programming.



- Why was process communication so complicated?
- Because most Computer Scientists were thinking in terms of sequential programming.



- It is different if the application in itself is concurrent.



1980

Mike Williams, Göran Båge, Seved Torsten-dahl and Bjarne Däcker proposed to create a Computer Science Laboratory

And were allowed to start on a small scale …

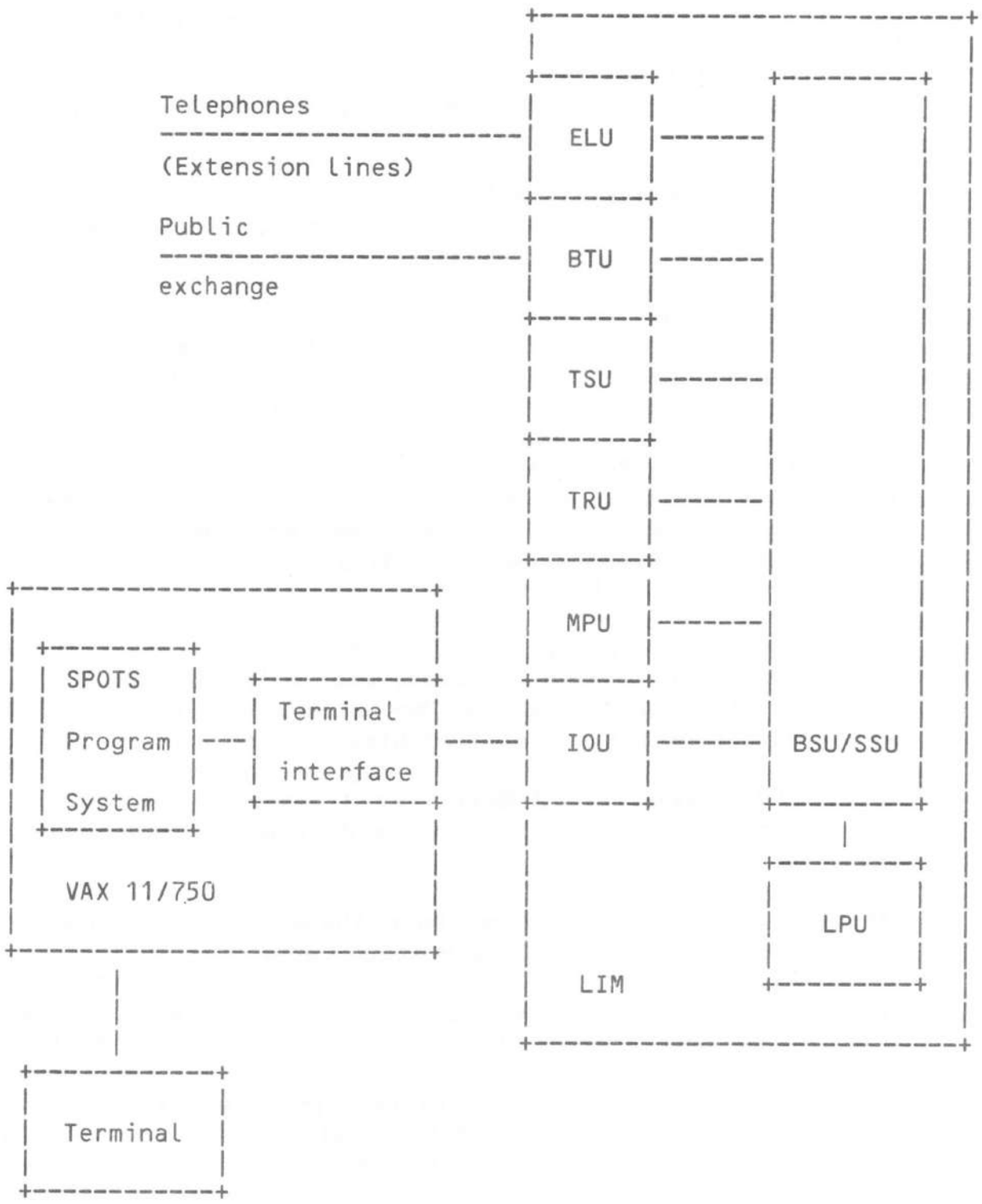
CSLab

Started early 1980’s with a mission to

1. Create a software technology for future telecom and support systems
2. Help introduce new technology in existing systems

ANSVARSOMRÅDE

XT/DU Datalogi har som ansvar på längre sikt att bygga upp en grundteknik inom programvaruområdet inför framtida telekom-system och stödsystem samt på kortare sikt att bidraga till introduktion av ny programvaruteknik i existerande system.



Laboratory setup

A telephone exchange, LIM (*Line Interface Module*), controlled by a VAX from DEC under Unix

Systematic experiments in programming telephony using different programming technologies

- Ada
- Concurrent Euclid (for CHILL)
- PFL (Parallel Functional Language)
- LPL (Logic Programming Language)
- OPS4 (Rule Based System)
- Frames (Object Oriented System)
- CLU (Abstract Objects)

It is becoming obvious that future telecommunication systems cannot be programmed with one language using one methodology. Future systems will probably be built up using many of the techniques used in these experiments. For example expert system technology might be used for the maintenance functions and the man machine interface, logic programming might be suitable for programming the signal system interfaces and parts of traffic handling and the underlying operating system might be programmed in an advanced imperative language.

It is becoming obvious that future telecommunication systems cannot be programmed with one language using one methodology. Future systems will probably be built up using many of the techniques used in these experiments. For example expert system technology might be used for the maintenance functions and the man machine interface, logic programming might be suitable for programming the signal system interfaces and parts of traffic handling and the underlying operating system might be programmed in an advanced imperative language.

Still confused but at a higher level …

It is becoming obvious that future telecommunication systems cannot be programmed with one language using one methodology. Future systems will probably be built up using many of the techniques used in these experiments. For example expert system technology might be used for the maintenance functions and the man machine interface, logic programming might be suitable for programming the signal system interfaces and parts of traffic handling and the underlying operating system might be programmed in an advanced imperative language.

This led to another round of experiments. Now also Joe and Robert joined. And a serious user group.

Erlang Design Team

Joe – Robert – Mike

At a workshop in February 1991

We did not intend to invent a new programming language. Perhaps those are created by international committees like CHILL and Ada?

But that is what happened. What about a suitable name like EriPascal? Perhaps EriLang?

We did not intend to invent a new programming language. Perhaps those are created by international committees like CHILL and Ada?

But that is what happened. What about a suitable name like EriPascal? Perhaps EriLang?

Erlang is used in telephony as a measure of offered load or carried load on elements such as telephone circuits or telephone switching equipment. [Wikipedia]

A well-known term in telecoms. That settled it.

We did not intend to invent a new programming language. Perhaps those are created by international committees like CHILL and Ada?

But that is what happened. What about a suitable name like EriPascal? Perhaps EriLang?

Erlang is used in telephony as a measure of offered load or carried load on elements such as telephone circuits or telephone switching equipment. [Wikipedia]

A well-known term in telecoms. That settled it.

Pascal, Ada, Occam are also named after mathematicians.

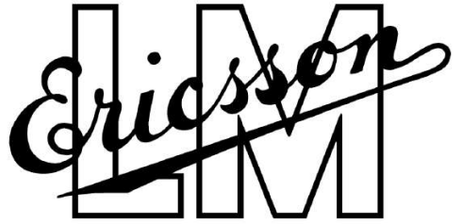
By now Ericsson had chosen a new logo

ERICSSON

By now Ericsson had chosen a new logo

ERICSSON

So we felt free to use the old logo



By now Ericsson had chosen a new logo

ERICSSON

So we felt free to use the old logo

Ta da … 🎺 🥁

The progress can be followed in the CSLab publications

The progress can be followed in the CSLab publications

Experiments with Programming Languages and Techniques for Telecommunications Applications. Eindhoven, April 14-18, 1986.

The progress can be followed in the CSLab publications

Experiments with Programming Languages and Techniques for Telecommunications Applications. Eindhoven, April 14-18, 1986.

The Phoning Philosphers' Problem or Logic Programming for Telecommunications Applications. Salt Lake City, September 23-26, 1986.

Using Prolog for Rapid Prototyping of Telecommunication Systems. Bournemouth, July 3-6, 1989.

The progress can be followed in the CSLab publications

Experiments with Programming Languages and Techniques for Telecommunications Applications. Eindhoven, April 14-18, 1986.

The Phoning Philosphers' Problem or Logic Programming for Telecommunications Applications. Salt Lake City, September 23-26, 1986.

Using Prolog for Rapid Prototyping of Telecommunication Systems. Bournemouth, July 3-6, 1989.

Erlang - An Experimental Telephony Programming Language.
International Switching Symposium.
Stockholm, May 27-June 1, 1990.

The progress can be followed in the CSLab publications

Experiments with Programming Languages and Techniques for Telecommunications Applications. Eindhoven, April 14-18, 1986.

The Phoning Philosphers' Problem or Logic Programming for Telecommunications Applications. Salt Lake City, September 23-26, 1986.

Using Prolog for Rapid Prototyping of Telecommunication Systems. Bournemouth, July 3-6, 1989.

Erlang - An Experimental Telephony Programming Language.
International Switching Symposium.
Stockholm, May 27-June 1, 1990.

Implementing a Functional Language for Highly Parallel Real Time Applications.
Florence, March 30-April 1, 1992.

Use of Prolog for Developing a new Programming Language.
London, England, April 1-3, 1992.

Prototyping Cordless Using Declarative Programming.
Yokohama, October 25-30, 1992.

The progress can be followed in the CSLab publications

Experiments with Programming Languages and Techniques for Telecommunications Applications. Eindhoven, April 14-18, 1986.

The Phoning Philosphers' Problem or Logic Programming for Telecommunications Applications. Salt Lake City, September 23-26, 1986.

Using Prolog for Rapid Prototyping of Telecommunication Systems. Bournemouth, July 3-6, 1989.

Erlang - An Experimental Telephony Programming Language.
International Switching Symposium.
Stockholm, May 27-June 1, 1990.

Implementing a Functional Language for Highly Parallel Real Time Applications.
Florence, March 30-April 1, 1992.

Use of Prolog for Developing a new Programming Language.
London, England, April 1-3, 1992.

Prototyping Cordless Using Declarative Programming.
Yokohama, October 25-30, 1992.

Turbo Erlang: An Efficient Implementation of a Concurrent Programming Language.
Madrid, May 24-25, 1993.

The progress can be followed in the CSLab publications

Experiments with Programming Languages and Techniques for Telecommunications Applications. Eindhoven, April 14-18, 1986.

The Phoning Philosphers' Problem or Logic Programming for Telecommunications Applications. Salt Lake City, September 23-26, 1986.

Using Prolog for Rapid Prototyping of Telecommunication Systems. Bournemouth, July 3-6, 1989.

Erlang - An Experimental Telephony Programming Language. International Switching Symposium. Stockholm, May 27-June 1, 1990.

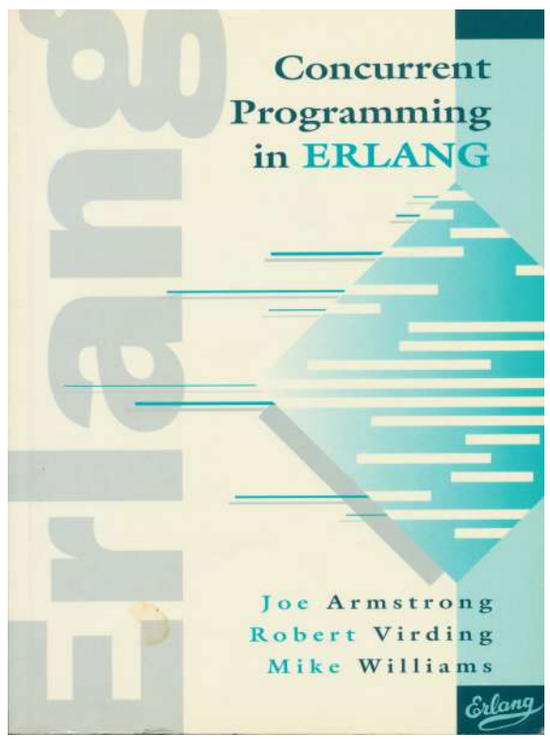
Implementing a Functional Language for Highly Parallel Real Time Applications. Florence, March 30-April 1, 1992.

Use of Prolog for Developing a new Programming Language. London, England, April 1-3, 1992.

Prototyping Cordless Using Declarative Programming. Yokohama, October 25-30, 1992.

Turbo Erlang: An Efficient Implementation of a Concurrent Programming Language. Madrid, May 24-25, 1993.

Concurrent Programming in Erlang. Prentice Hall, 1993.



Three wise guys. Joe, Robert, Mike at Bellcore, December 1989

Thank you

http://www.cs-lab.org/