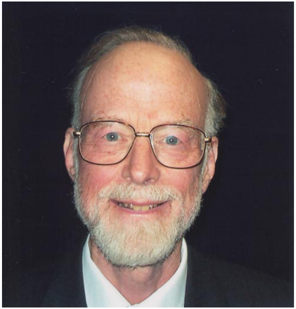
Concurrent Object Oriented Languages

Synchronous Message Passing

https://wiki.cse.yorku.ca/course/6490A

Communicating Sequential Processes (CSP)

C.A.R. Hoare. Communicating sequential processes. *Communications of the ACM*, 21(8):666-677, August 1978.

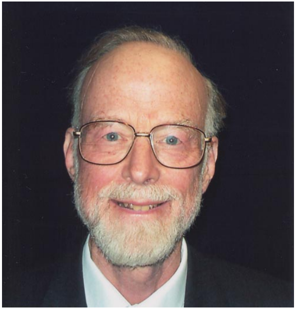


sir Charles Antony Richard (Tony) Hoare

source: cs.ox.ac.uk

Communicating Sequential Processes (CSP)

C.A.R. Hoare. *Communicating Sequential Processes*. 1985.



sir Charles Antony Richard (Tony) Hoare

source: cs.ox.ac.uk

Syntax of CSP

CSP has static process creation.

$$\texttt{[name :: command} \parallel \cdots \parallel \texttt{name :: command]}$$

Syntax of CSP

CSP uses synchronous message passing to communicate.

- Receive command

  ```
  name?pattern
  ```

- Send command

  ```
  name!expression
  ```

Communication in CSP

Question

What is the result of the following communication?

```
[ sender :: receiver!(1,2)
  ‖ receiver :: sender?(1,x) ]
```

Answer

The variable x is assigned the value 2.

Communication in CSP

Question

What is the result of the following communication?

```
[ sender :: receiver!(1,2)
  ‖ receiver :: sender?(3,x) ]
```

Answer

No communication takes place since the expression (1,2) does not match the pattern (3,*x*).

Syntax of CSP

Conditional command

```
[ guard → command □ ··· □ guard → command ]
```

guard

- Boolean expression
- receive command
- Boolean expression ; receive command

Syntax of CSP

Iteration command

$*[\ \texttt{guard} \rightarrow \texttt{command} \ \square \cdots \square \ \texttt{guard} \rightarrow \texttt{command}\ ]$

guard

- Boolean expression
- receive command
- Boolean expression ; receive command

Examples in CSP

Express a semaphore, named `semaphore`, and a process, named `process`, using that semaphore to protect its critical section in CSP.

Examples in CSP

Express the consumer-producer problem in CSP. The producer, named `producer`, produces the integers 1, . . . , 100 and the consumer, named `consumer`, prints the integers it consumes. Both interact with the buffer, named `buffer`.

Examples in CSP

Let

```
reader(i) ::
  *[ scheduler!request();
     read();
     scheduler!done() ]

writer(i) ::
  *[ scheduler!request();
     write();
     scheduler!done() ]
```

Implement `scheduler` to solve the readers-writers problem.

Examples in CSP

What is wrong with

```
phil(i) ::
  *[ THINK;
     fork(i)!pickup(); fork((i+1) mod N)!pickup();
     EAT;
     fork(i)!putdown(); fork((i+1) mod N)!putdown()

fork(i) ::
  *[ phil(i)?pickup()
       → phil(i)?putdown()
     □ phil((i−1) mod N)?pickup()
       → phil((i−1) mod N)?putdown() ]
```

Examples in CSP

The sieve of Eratosthenes is a simple, ancient algorithm for finding all prime numbers up to a specified integer.

Eratosthenes

source: world.mathigon.org

Examples in CSP

Processes:

- `generator` that generates 2, 3, . . .
- `sieve(i)`, for $1 \leq i \leq n$, where n is the number of primes to be generated (`sieve(n)` is defined differently).

Examples in CSP

```
sieve(0) ::
  n = 2;
  *[ sieve(1)!n; n = n + 1 ]

sieve(i) ::
  sieve(i - 1)?p;
  print(p);
  *[ sieve(i - 1)?n
       → [ n mod p == 0 → skip
          □ n mod p != 0 → sieve(i + 1)!n ]

sieve(100) ::
  sieve(99)?p; print(p)
```

Assignment 1

Due: October 1

Presentations: October 8 and 13