Non-determinism by design

A property of a computation which may have more than one result.

Non-determinism by interaction

A property of the operation environment which may lead to different sequences of (concurrent) stimuli.

Selective waiting in Occam2

- Guarded commands
- Non-determinism by design
- Non-determinism by interaction
- Basic forms of selective synchronization

References for this chapter

If the call is not accepted immediately

The alternative is chosen.

Example:

Controller.Request (Some_Item);

---

Correctness of non-deterministic programs:

Brief correctness:

$P(O) \Rightarrow \text{terminates} (\text{Program}(O), O) \Rightarrow Q(O, O)$

Safety properties:

$P(O) \Rightarrow \text{processes} (Q, O) \Rightarrow Q(O, O)$

Licences properties:

$P(O) \Rightarrow \text{processes} (Q, O) \Rightarrow Q(O, O)$

---

Correctness of non-deterministic programs:

- Correctness predicates need to hold true irrespective of the actual sequence of interaction points.
- Correctness predicates need to hold true for all possible sequences of interaction points.
- These correctness properties are not the basis for implementation, i.e., they are not for specifying and checking correct programs.
- These correctness properties are used to analyze and reason about potential system properties, i.e., supporting the correctness of individual programs.

---

Non-determinism:

- Sources of non-determinism:
  - Source code by design
  - Source code by interaction
  - Source code by synchronization
  - Source code by selection

---

Summary:

- Non-determinism by design
  - Systematic writing
  - Pathological writing

- Non-determinism by interaction
  - Selective synchronization
  - Timed entry-calls
  - Asynchronous selection

- Correctness of non-deterministic programs:
  - Sources of non-determinism
  - Functions & libraries