

0.1 Checking safety property using CSP

The history can be captured as a trace of a system. In addition to performing the function body, each function call sends a **Call** event before the function body and a **Return** event after the function body. Figure 1 is the definition of **Call** and **Return** channel in CSP. The definition of the channel usually need to include the identity of the calling process, the function called, and its parameter.

```
--identity of the calling process
--function called by the process
channel Call : TypeThreadID . TypeOps
--identity of the calling process
--function called by the process
--return value of the function call
channel Return: TypeThreadID . TypeOps . TypeThreadID
```

Figure 1: Definition of call

To check the safety property, we check that a testing system built from some processes using the concurrent datatype refines a specification process built from the object definition in CSP trace model.

A generic and scalable system is used for the testing system to generate possible histories of processes using concrete datatype. Each process in the testing system can call any function from the concurrent object with any arguments allowed. Such construction allows to build a system with an arbitrary number of processes, and cover as many histories as possible. The specification generates all valid histories. The process uses the same number of linearizer processes synchronizing on events from the **Sync** channel. Each linearizer process repeatedly calls a function, synchronize with zero or some processes, and returns according to the calling argument and extra information from the synchronization point. We shall see a concrete testing system and specification process in the MenWomen section.

0.2 Checking liveness property using CSP

For liveness property, we check the same generic and scalable testing system refines the same specification process, but in the failure model. Suppose all process calls **manSync**. Since a linearizer process calling **manSync** sends **Return** event only after synchronizing **Sync** event with another linearizer process calling **womanSync**, the linearizer will refuse to return any function call, which is a expected behavior. One can use a datatype-specific specification process that does not explicitly use any synchronization points. However, reusing the linearizer process is easier.