### 6.3.3 Compatibility of component types

Type compatibility of component types has to be considered in different conditions:

1. Compatibility of a component reference value with a component type (e.g. when passing a component reference as an actual parameter to a function or an altstep or when assigning a component reference value to a variable of different component type): a component reference "b" of component type "B" is compatible with component type "A" if all definitions of "A" have identical definitions in "B".
2. Runs on compatibility: a function or altstep referring to component type "A" in its runs on clause may be called or started on a component instance of type 'B' if all the definitions of "A" have identical definitions in "B".
3. Mtc compatibility: a function or altstep referring to component type "A" in its mtc clause may be called or started in any context that has a mtc clause of type "B" or a testcase with a runs on clause of type "B" if all the port definitions of "A" have identical definitions in "B". If the type of the mtc is unknown in the calling function, this can lead to runtime errors if the component type "A" is not mtc-compatible with the type of the running mtc.
4. System compatibility: a function or altstep referring to component type "A" in its system clause may be called or started in any context that has a system clause of type "B" or a test case with a runs on clause of type "B" and no system clause if all the port definitions of "A" have identical definitions in "B". If the type of the system is unknown in the calling function, this can lead to runtime errors if the component type "A" is not system-compatible with the type of the system the current test case was started on.

Identity of definitions in "A" with definitions of "B" is determined based on the following rules:

a) For port instances, both the type and the identifier shall be identical.

b) For timer instances, identifiers shall be identical and either both shall have identical initial durations or both shall have no initial duration.

c) For variable instances and constant definitions, the identifiers, the types and initialization values shall be identical (in case of variables this means that either the values are missing in both definitions or are the same).

d) For local template definitions, the identifiers, the types, the formal parameter lists and the assigned template or template field values shall be identical.

e) The visibility of compatible component types shall be identical.