**Rational.** Statemate

IBM

Check Model Guide

# Rational Statemate Check Model Guide

Before using the information in this manual, be sure to read the "Notices" section of the Help or the PDF file available from **Help > List of Books**.

# Contents

## Table of Contents

# Using the Check Model Tool

The IBM® Rational® Statemate®Check Model tool is an error-handling tool that enables you to check for errors and warnings in Rational Statemate models. While designing model charts and diagrams, the Rational Statemate graphical editors handle simple, syntactical errors. Use Check Model to both detect and resolve more complex errors in relational elements of models.

Check Model performs the following types of checks:

- ◆ **Correctness** - Checks for inconsistencies in the model. Correctness checks return errors.

- ◆ **Completeness** - Checks for redundancy and incompleteness in the model. Completeness checks return warnings.

- ◆ **Design** - Checks for design problems in the model. This type of check is available in Rational Statemate extended mode. Design checks return warnings.

- ◆ **Rational Statemate MicroC Compatibility** - Reports on Rational Statemate features used that are not supported in MicroC code generation.

Correctness checks detect illegal situations regarded as errors, whereas most completeness checks warn of possible oversights in your System Under Design specifications. Some completeness warnings might have minimal or no impact on a model's overall design or the behavior of a system.

This section describes how to check Rational Statemate models using the Check Model tool. The topics are as follows:

- ◆ **Performing a Check on a Single Chart**

- ◆ **Creating Check Model Profiles**

- ◆ **Customizing Check Model Profiles**

- ◆ **Analyzing Results and Fixing Errors**

- ◆ **Enabling and Disabling Checks**

- ◆ **Check Model Test Settings File**

# Performing a Check on a Single Chart

At any stage during your work on an individual chart, you can easily run Check Model. Best practices advise running Check Model frequently during the design process to ensure that errors are caught before models become too complex and errors compound.

To check an individual chart, do the following:

1. Open the chart within its graphic editor.

2. Select **Tools** > **Check Model**.

Check Model runs a check on the chart and all its elements and reports the number of errors in correctness or warnings in completeness in the **Messages** field. In addition, a list of check names and numbers for erroneous elements open in a text editor.

# Creating Check Model Profiles

Check Model enables you to create a configuration of charts and diagrams to check and recheck by using profiles. Each Check Model profile displays the scope of all the charts and diagrams to be checked.

To create a Check Model profile, do the following:

1. Open the Check Model tool, then select **File** > **New Profile**. The New Check Model Profile dialog box opens.

2. Enter a **Name** for the profile, then click **OK**.

3. Click **Add Selected Chart with Descendants to Profile** . The Charts Tree dialog box opens. See the following figure.

4. Select the **names** of the charts you want to check in the chart tree. To include additional charts within a profile's scope, even if those charts are not hierarchically related, select the chart names from the chart tree.

5. Click **OK**. The Charts Tree dialog box closes.

6. Select **File** > **Save.**
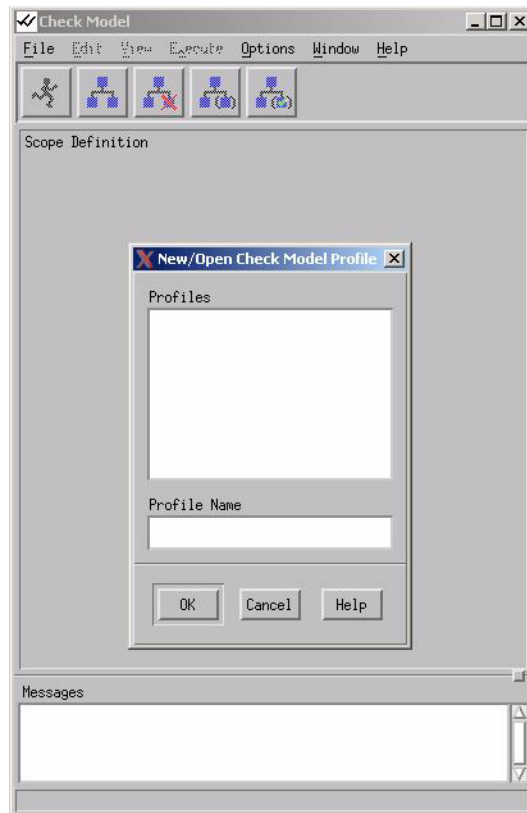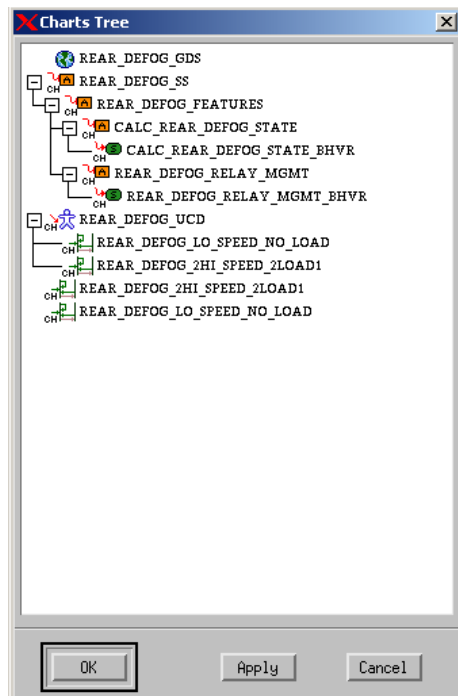
# Customizing Check Model Profiles

To delete charts from a profile's scope, select the chart name and click **Remove Selected Objects from Profile** ![icon]. The profile editor removes the selected chart and all its children charts from the profile.

## Exclude from Check Model Scope

Exclude from Scope allows trimming of the activity chart's hierarchy at a defined activity chart. The result is a scope that includes only a part of the model hierarchy.

The activity chart that was excluded from the scope is marked with a special icon and does not show any of its descendants.

To exclude and activity chart from the scope, select **Edit > Exclude from Scope**.

To include the activity chart back into the scope, select **Edit > Include in Scope**.

# Performing a Check on a Profile

Rational Statemate enables you to check multiple charts and diagrams, and the relation of their elements, using the Check Model profile.

To perform a check on a configuration of charts in a model, do the following:

1. Create a new profile or open a saved profile of the charts you want to check.
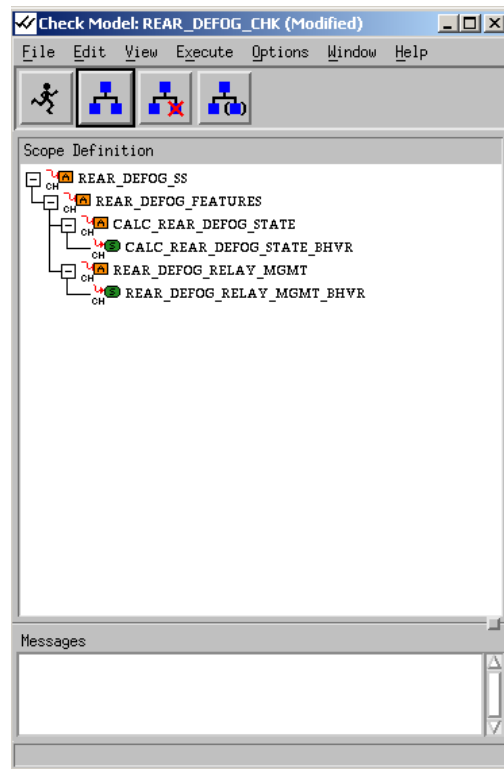


2. Click **Execute Current Check Model Profile** .

   The Check Model Results window opens (see the following figure) and reports correctness errors and completeness warnings for specific elements, listed by check number and name.

   To sort the results, select **View** > **Sort**.
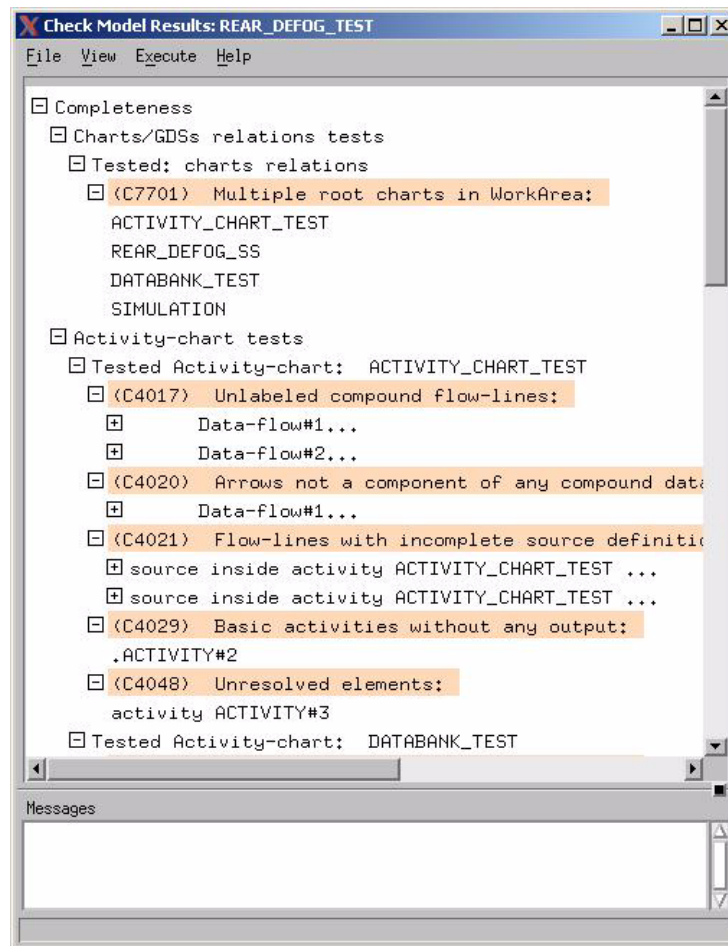
# Analyzing Results and Fixing Errors

The **Check Model Results** window reports correctness errors and completeness warnings for model elements by check number and name. After running a check, Check Model lists the erroneous element and its location in the model for each error.

You can examine and edit the properties of these erroneous elements through the Check Model Results window by right-clicking on the element name and clicking **Properties** or opening the graphic editor or property in which it appears.
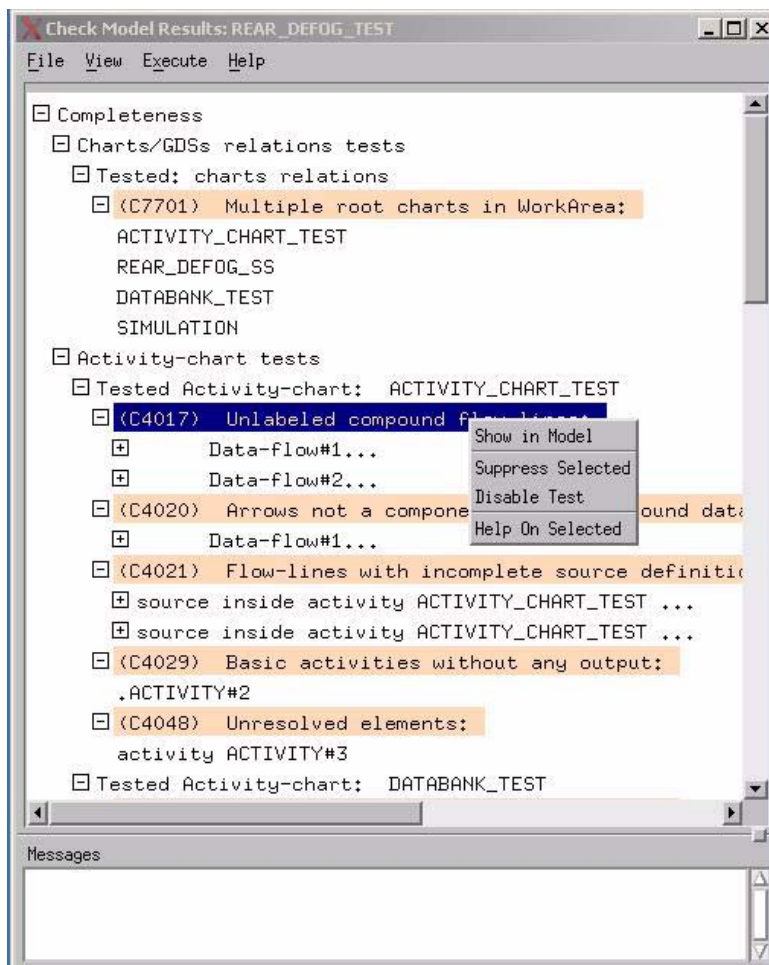


For descriptions of each Check Model check by chart and chart relation, see the following sections:

- ◆ **Charts and GDS Relation Checks**

- **Statechart Checks**
- **Activity Chart Checks**
- **Module Chart Checks**
- **Global Definition Set Checks**
- **Sequence Diagram Checks**
- **Statechart and Activity Chart Relation Checks**
- **Activity and Module Chart Relation Checks**

# Automatic Population of Info-Flow

This feature enables a quick resolution for the following check-model errors, using the check-model suggested actions mechanism:

- (C4071) Elements affected by an activity but are neither used internally nor output
- (C4073) Elements that are used by an activity but are neither input nor affected internally

When the tool is able to suggest an information-flow path resolving these errors, a new default action is offered on the error node, called Populate Information Flow(s).

To open the Populate Information Flow dialog box, right-click on the error node in the Check-Model tree view and select **Populate Information Flow(s) > Set Information Flow Path**. This dialog box enables you to define alternate information-flow paths to resolve the check-model error. When the dialog box opens, it displays the first possible path that resolves the error. You can modify this suggested information-flow path or select an alternate route.
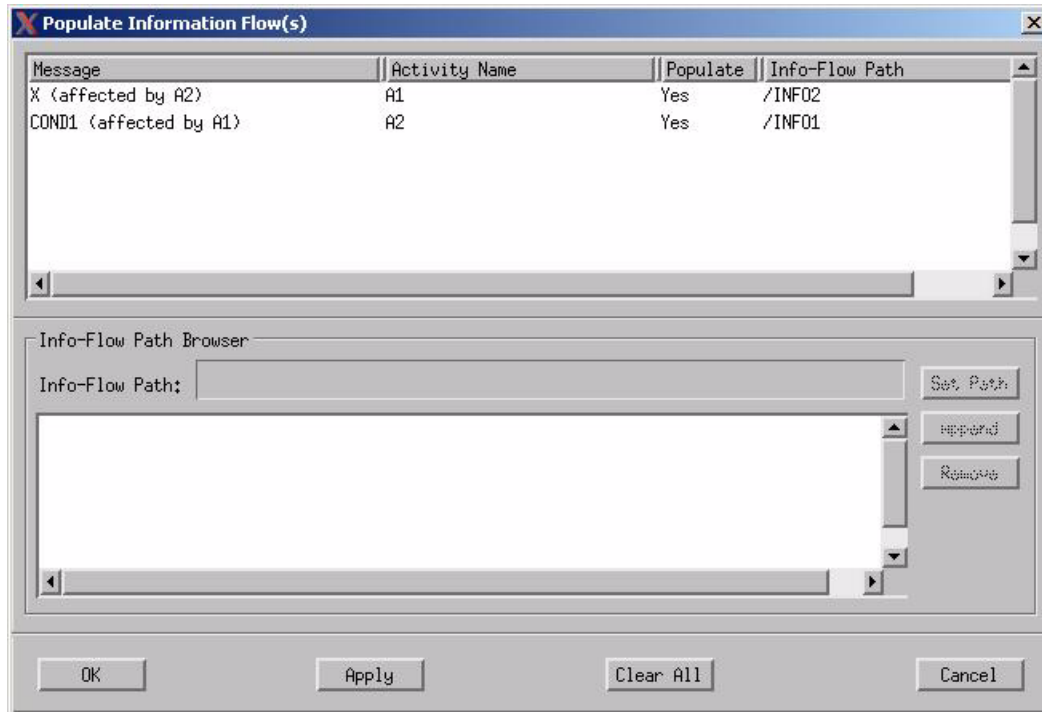
To apply the default, suggested path, select **Populate Information Flow(s) > Set Default Path**.

## Suppressing Check Model Checks

To further isolate which errors the tool reports, you can suppress selected errors from the report as follows:

1. Execute a check on a model profile.

2. Right-click the check name or erroneous error, then select **Suppress Selected**.

Check Model runs an analysis on suppressed checks, but does not report them. Suppressed errors remain not visible in the report until you unsuppress them.

To view suppressed errors, select **View** > **Messages Suppression View** from the Check Model Results window (see the following figure).

To unsuppress a check or error, right-click and select **Unsuppress Selected**.

```
┌──────────────────────────────────────────────────────┐
│ X Messages Suppression View: REAR_DEFOG_CHK    _ □ ✕  │
├──────────────────────────────────────────────────────┤
│ File  View  Execute  Help                            │
├──────────────────────────────────────────────────────┤
│ ⊟ Correctness                                        │
│   ⊟ Activity-chart tests                             │
│     ⊟ Tested Activity-chart:  REAR_DEFOG_SS          │
│       ⊟ (C3053)  Elements that flow to/from instance with cont│
│         ⊞ RDEFOG_LED flows from REAR_DEFOG_FEATURES...│
│                                                        │
│                                                        │
│                                                        │
│                                                        │
│                                                        │
│                                                        │
│                                                        │
│ ◄                                                   ► │
├──────────────────────────────────────────────────────┤
│ Messages                                             │
│                                                      △│
│                                                      ▽│
└──────────────────────────────────────────────────────┘
```
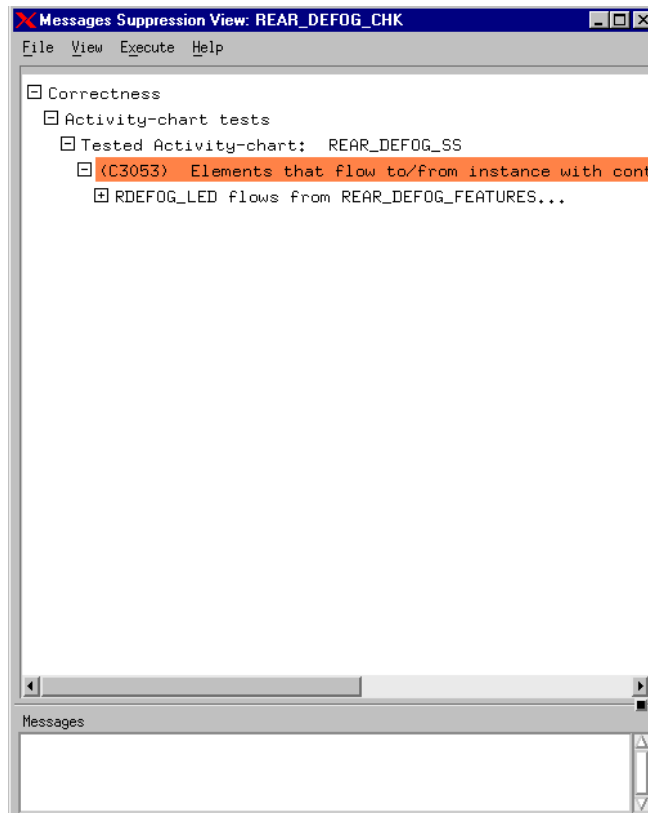
# Enabling and Disabling Checks

To refine the focus of a scan for errors in a model, the Check Model tool provides the option of configuring which checks run in a given analysis. You can enable or disable any Check Model check, either for specified profiles or globally, for checks of all model elements within a project, by a user, or on a system.

## Changing Settings for a Specific Profile

To set check options for a specific profile:

1. Open the profile in Check Model, then select **Options** > **Test Settings**. The Test Settings dialog box opens for the profile, as shown in the following figure.
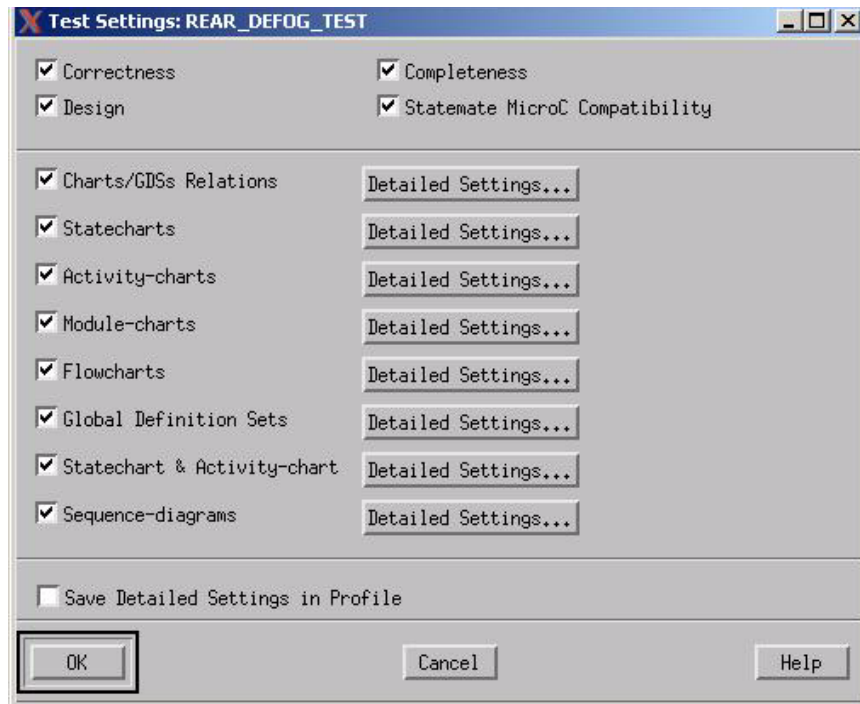


2. Select the **type of check** to run and the **type of charts** to check.

3. Disable and enable specific checks by clicking **Detailed Settings**, then selecting **Yes** or **No** in the **Value** column.
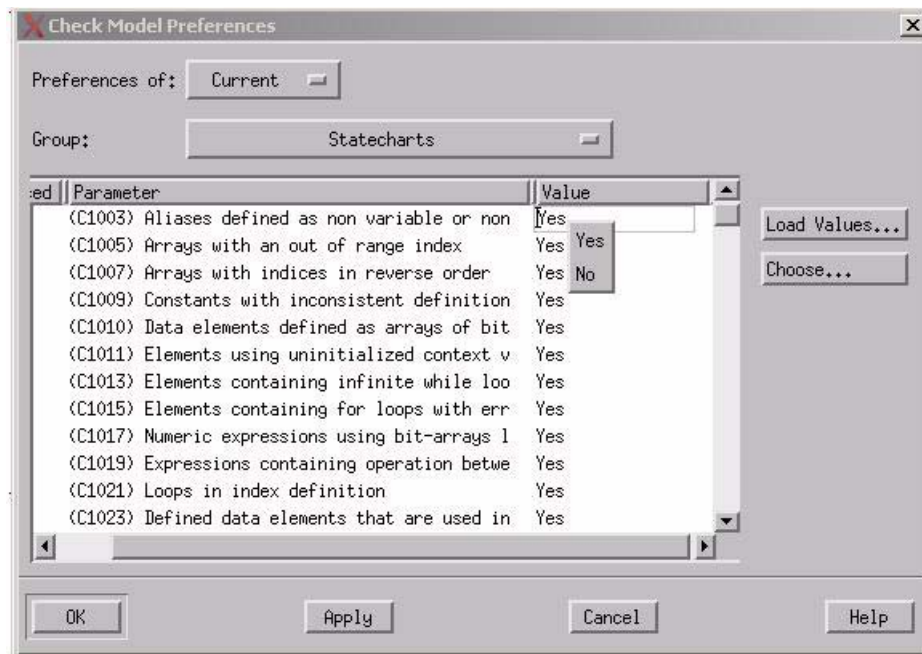
4.  Click **OK** to save your changes and close the dialog box.

5.  Select **Save Detailed Settings In Profile** to save the preferences for this specific profile without changing the preference settings in the User Check Model preferences.

6.  Click **OK** to dismiss the Test Settings dialog box.

# Changing Global Settings

To set check preferences globally for all charts and diagrams:

1. From the Check Model profile, select **Options** > **Preferences Management**.

   Alternatively, from the Rational Statemate browser, select **Options** > **Preferences Management** > **Check Model** to open the Check Model Preferences dialog box.

2. Select the environment for the configuration of your check preferences in the **Preferences of** drop-down menu. The possible options are as follows:

   ◆ **System** - Retrieves the configuration for all checks set on a system

   ◆ **Project** - Retrieves the configuration for all checks set within the currently selected project

   ◆ **User** - Sets the configuration for all checks set by a user

   ◆ **Default** - Resets the default configuration for all checks

   ◆ **Current** - Displays the current preferences of the active window session

   **Note:** **System** and **Project** settings are permission-sensitive.

3. Select the chart configuration you want to display from the Group drop-down menu

4. Enable or disable checks in the Value column.

5. Click **Apply**, then **OK**.

---
**Note**
---

System and Project preferences can be enforced (values cannot be changed by the user). This information is displayed in the Enforced column.

## General Check Model Preferences

Check Model also provides a means so that the user can control some preferences that are not specific to any checks.

To access these preferences, select from the Main window **Project > Preference Management > Check Model**.
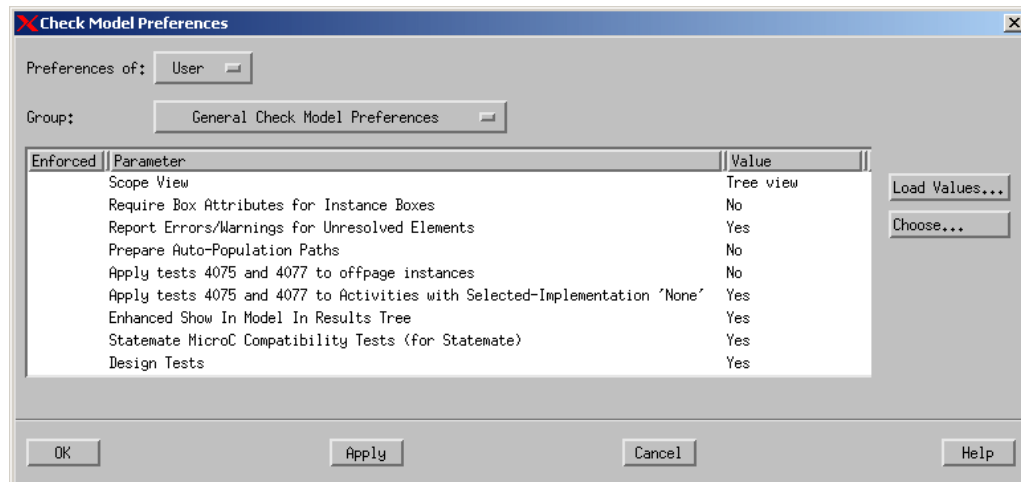


- ◆ **Scope View** controls the method that test results are displayed to the screen.
- ◆ **Require Box Attributes for Instance Boxes**:
  - – Set to **No** when you want to eliminate the Check Model tests 6005, 4005, 2003, 17003 ("Elements with incorrect or incomplete attribute definitions") for instance boxes.
  - – Set to **Yes** for the Check Model tool to check for box attributes also in instance boxes.
- ◆ **Report Errors/Warnings for Unresolved Elements**:
  - – Set to **Yes** when you want the Check Model tool to report various Warnings and Errors regarding the element, according to the selected Check Model checks.
  - – Set to **No** when you want the Check Model tool to report only those elements that are unresolved. No other Warnings or Errors are generated for the unresolved elements.
- ◆ **Prepare Auto-Population Paths** controls the calculation of the C4073/C4071 default-action (calculation of possible information flow paths) with a preference. In order to enable the feature of automatic population, set to **Yes**. The default value is **No**.

- ◆ **Apply tests 4075 and 4077 to offpage instances** controls whether or not to expand the following tests also for nonbasic activities (off-page):

    – (C4075) Elements output from activity that are not affected (generated or modified) internally.

    – (C4077) Elements input to activity that are not used internally.

- ◆ Apply tests 4075 and 4077 to Activities with Selected Implementation 'None' controls the execution of Check Model tests 4075 and 4077 for activities with Selected-Implementation 'None'. When set to No, tests 4075 and 4077 will not check activities with selected implementation 'None'. The default is Yes.

- ◆ **Enhanced Show In Model In Results Tree** controls the availability of the **Show in Model** option from the Check Model results tree for an enhanced list of tests.

- ◆ Rational Statemate MicroC Compatibility Tests (for Rational Statemate) controls the execution of Rational Statemate MicroC compatibility tests in Rational Statemate Classic. The default is No.

- ◆ Design Tests controls the execution of the "Design" tests of the Check Model tool. A new Check Model profile reads this preference and uses its value as the default value for the Design check box on the Test Settings dialog box (to open, select Options > Test Settings). The default is Yes.

# Check Model Test Settings File

You can create and use Check-Model test-sets. Test-sets are files that define enabled/disabled Check-Model tests that are saved in the Rational Statemate Preferences directory and are available for all users using that directory.

To manage test setting files:

1.  Choose **Tools > Check Model** to open the Check Model dialog box and the New/Open Check Model Profile dialog box.

2.  On the New/Open Check Model Profile dialog box, select a profile or create a new one.

3.  On the Check Model dialog box, choose **Options > Test Setting Configurations** to open the Test Settings Configuration dialog box, as shown in the following figure:
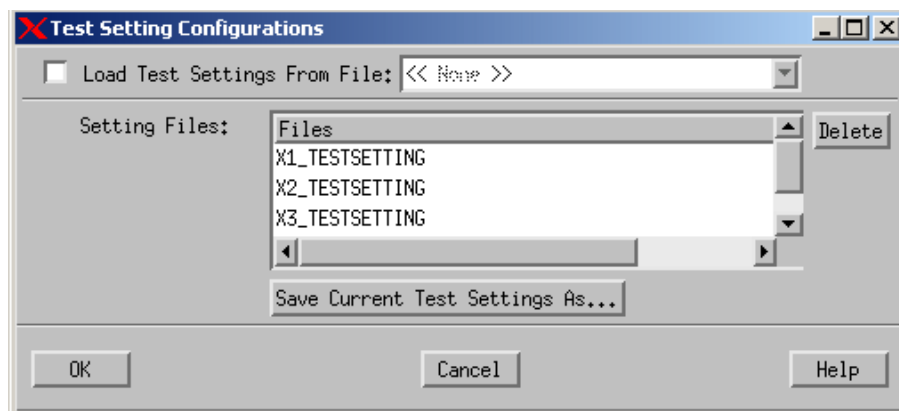


4.  Depending on what you want to do, you can:

    ◆  Choose a test setting file from which to load test settings for the current session; click the **Load Test Settings From File** check box.

    ◆  Save the current test settings (as defined on the Test Setting dialog box, which you open by choosing **Options > Test Settings**) as a test settings file; click the **Save Current Test Settings As** button.

    ◆  Delete a test setting file; click **Delete**.

# Charts and GDS Relation Checks

This section provides the Check Model checks for chart and Global Definition Set (GDS) relations. The checks are listed in numerical order:

## (C7601) Loop in chart usage

There are generic/off-page charts whose usage of each other causes a loop.

**Error In:**

Correctness

**Example:**

AAA is an off-page chart used in a generic chart GGG; BBB is an off-page chart used in chart AAA. GGG (generic) is instantiated in chart BBB.

**Erroneous Elements:**

All charts in the loop

## (C7603) Illegal usage type of charts

Charts that exist in the model are used as charts of another type.

**Error In:**

Correctness

**Example:**

Activity-chart AAA is used as an off-page chart of a state S@AAA. Generic chart GGG is used as an off-page chart S@GGG.

**Erroneous Elements:**

Charts

# (C7605) Multiple offpage instances

There are charts referenced as off-page charts of more than one box.

**Error In:**

Correctness

**Example:**

sss is a statechart. There are two states in the model using this statechart as an off-page chart, S1@SSS and S2@SSS.

**Erroneous Elements:**

Charts

# (C7607) Inconsistent usage of chart

Charts that are not defined in the model are used inconsistently, both as off-page and generic, or in one place as a statechart and in another as an activity-chart.

**Error In:**

Correctness

**Example:**

sss does not exist in the model. There are two states in the model: S1@SSS and S2<SSS.

AAA does not exist in the model. There is a control activity CTRL@AAA (AAA is used as a statechart) and a regular activity A1@AAA (AAA is used as an activity-chart).

**Erroneous Elements:**

Charts

# (C7609) Inconsistency of parameter bindings in different generic instances

Instance boxes of the same generic chart have an inconsistent binding to the same formal parameter.

**Error In:**

Correctness

**Example:**

A<GGG and B<GGG are two instances of the generic GGG.

- ◆ In A, event E is bound to formal parameter P1 and in B, data-item D is bound to P1.
- ◆ In A, event G is bound to P2 and in B, there is no actual binding to P2.

**Erroneous Elements:**

Instance boxes

# (C7611) Mutual references in global definition sets

Global Definition Sets (GDSs) have references to elements in other global definition sets (in a "circular" manner).

**Error In:**

Correctness

**Example:**

GLOBAL_TYPES is a GDS that contains the following definitions:

- ◆ BIG_ARRAY is an integer array from 1 to K.
- ◆ MY_INT is an integer from 1 to 10.

CONSTANTS is a GDS that contains the following definitions:

- ◆ K is defined as a constant integer equal to 5.
- ◆ M is defined to be of type MY_INT.

GLOBAL_TYPES and CONSTANTS are two global definition sets that have mutual references.

**Erroneous Elements:**

Global definition sets

# (C7701) Multiple root charts in Workarea

The workarea contains multiple regular charts (non-generic) that are *not* used as off-page charts of other boxes. Therefore, the static structure of charts has multiple roots.

**Error In:**

Completeness

**Erroneous Elements:**

Charts

# (C7703) Missing charts in Workarea

Charts are referenced as generics or off-page, but are not defined in the workarea.

**Error In:**

Completeness

**Erroneous Elements:**

Charts

# (C7705) Multiple subroutines with the same name

Subroutines are defined in different charts with the same name.

**Error In:**

Completeness

**Format:**

The format is as follows:

```
<subroutine name>
<Task/Procedure/Function> <chart name>:<subroutine name>
```

**Example:**

```
MULT
Function SCH:MULT
Function SCH_ADA:MULT
```

**Erroneous Elements:**

Subroutines

# (C7707) Procedural charts not connected to any subroutine

Procedural statecharts/flowcharts are not connected to any subroutine.

**Error In:**

Completeness

**Erroneous Elements:**

Charts

# (C7709) Procedural charts connected to more than one subroutine

Procedural statecharts/flowcharts are connected to more than one subroutine.

**Error In:**

Completeness

**Erroneous Elements:**

Charts

# Statechart Checks

This section provides the Check Model checks for statecharts. The checks are listed in numerical order.

## (C1001) Loops in element definition

Textual elements use themselves in their own definition, via a "chain" definition.

**Error In:**

Correctness

**Example:**

Compound event **E1** is defined as `'E2 or E3'`; E2 is defined as `'E1 or E4'`.

**Erroneous Elements:**

All textual elements in the loop

## (C1003) Aliases defined as non variable

Aliases are illegally defined because the element appearing in the definition of the alias is defined as a non-variable compound or constant, or it is not a bit-array.

**Error In:**

Correctness

**Erroneous Elements:**

Data-items

# (C1005) Arrays with an out of range index

Arrays or bit-arrays are used with an index value that is out of the defined index range.

**Error In:**

Correctness

**Example:**

A is an integer array from 1 to 100. The model contains the following action:

```
A(101)=1
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C1007) Arrays with indices in reverse order

There are arrays or bit-arrays for which there are references to slices whose borders are in a reverse order relative to their type definition.

**Error In:**

Correctness

**Example:**

A is a bit-array from 31 down to 0. The model contains the following action:

```
A(0..7)=3
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C1009) Constants with inconsistent definition

There are either constant data-items or conditions whose values do not match their type definition; constant strings, arrays, or integers whose length does not match their defined length; or a constant integer whose value is not in the defined value range.

**Error In:**

Correctness

**Erroneous Elements:**

Data-item or condition

# (C1010) Data elements defined as arrays of bit

Elements are defined as array of a user-defined type (UDT), which is defined as bit.

**Error In:**

Correctness

**Example:**

`DI1` defined as array of `BIT1`, which is defined as bit.

**Erroneous Elements:**

Data-items, UDTs, and fields

# (C1011) Elements using uninitialized context variables

Elements contain action expressions that use an uninitialized context variable.

**Error In:**

Correctness

**Example:**

There is a transition labeled with the following label:

```
E / Y=$X; $X=Z
```

$X is used before any value is assigned to it.

**Erroneous Elements:**

States, transitions, and actions

# (C1013) Elements containing infinite while loops

This check detects cases in which the `while` loop control expression does not contain a context variable and there is no `break` statement in the loop.

**Error In:**

Correctness

**Erroneous Elements:**

States, transitions, and actions

# (C1015) Elements containing for loops with erroneous range indices

Elements contain action expressions in which there are `for` loop statements with invalid ranges.

**Error In:**

Correctness

**Example:**

There is an action that contains the following `for` loop statements:

```
for $I in 10 to 1 loop
...
for $I in 1 down to 10 loop
...
```

**Erroneous Elements:**

States, transitions, and actions

# (C1017) Numeric expressions using bit-arrays longer than 32 bits

There are elements containing numeric expressions that use bit-arrays longer than 32 bits. There is an automatic conversion from a bit-array to a numeric value only for a bit-array that is less than or equal to 32 bits.

**Error In:**

Correctness

**Erroneous Elements:**

States, transitions, and actions

# (C1019) Expressions containing operation between arrays with unequal length

Elements contain assignments or comparisons between arrays or array slices of different lengths.

**Error In:**

Correctness

**Example:**

There is an action that contains the assignment `A=B`, where `A` is defined as an array from 1 to 3, and `B` is an array from 1 to 5.

**Erroneous Elements:**

States, transitions, and actions

# (C1021) Loops in index definition

Data elements use constants in their definition in a "circular" manner.

**Error In:**

Correctness

**Example:**

`AR` is an array from 1 to `N`, while `N` is defined as `rindex (AR)`.

**Erroneous Elements:**

Data-items, conditions, events, and user-defined types

# (C1023) Defined data elements that are used incorrectly

Elements contain expressions in which data-items that are explicitly defined to be of some data-type are used in a way that does not match the type consistency and conversion rules.

**Error In:**

Correctness

**Example:**

I is defined as an integer; S is defined as a string. There is an action that contains the following assignment:

```
I=S
```

**Erroneous Elements:**

States, transitions, events, conditions, data-items, and actions

# (C1025) Data elements without type definition that are used inconsistently

There are unresolved or explicitly defined data-items with no type definition that are used in expressions in an inconsistent manner.

**Error In:**

Correctness

**Example:**

D is an unresolved element (it is not explicitly defined); I is an integer; S is a string. D appears in the following assignments:

```
D=I; D=S
```

**Erroneous Elements:**

Data-items, conditions, and user-defined types

# (C1027) Predefined Statemate functions that are used incorrectly

Predefined Rational Statemate functions are used with parameters of a wrong type, the wrong number of parameters, or the wrong return type.

**Error In:**

Correctness

**Erroneous Elements:**

Functions

# (C1029) Subroutines that are used incorrectly

User functions are used inconsistently in different expressions (for example, with different return type, different parameter type, or a different number of parameters).

> **Note:** When working with MicroC components, this test also reports situations where a subroutine is called with an actual 'out' parameter that is bound to an external symbol with get/set APIs.

**Error In:**

Correctness

**Erroneous Elements:**

Functions

# (C1031) Loops in compound transitions

A group of transition segments, linked through connectors, creates a loop (cycle) that does not contain a state.

**Error In:**

Correctness

**Erroneous Elements:**

Transition segments in the loops

# (C1033) Default transitions with illegal sources

Transitions have multiple sources and one of them is either a conditional connector or a history connector.

**Error In:**

Correctness

**Example:**

In the following figure, the compound transition entering S4 (whose source set consists of the default connector in S1 and the state S2) is an erroneous transition.
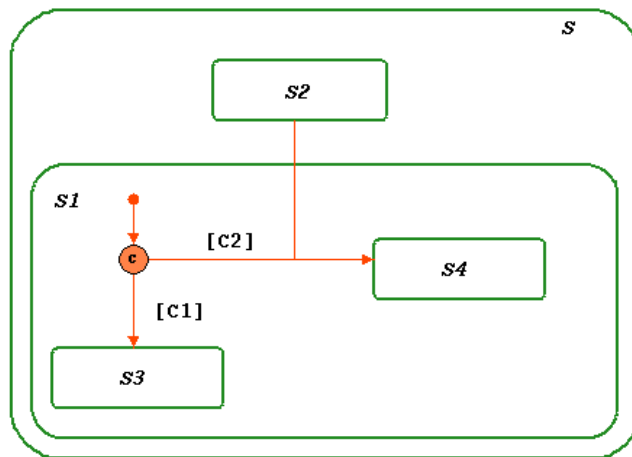


**Erroneous Elements:**

Components of the erroneous transitions

# (C1035) Default transitions with illegal targets

There are transitions whose source is a conditional connector or a history connector, and whose targets are not descendants of the state in which the source connector resides.

**Error In:**

Correctness

**Example:**

In the following figure, there are two possible default states when entering s1. One entrance leads to s3 (which is a substate of s1) and the other leads to s2 (which is not a descendant of s1 and is therefore an invalid transition).
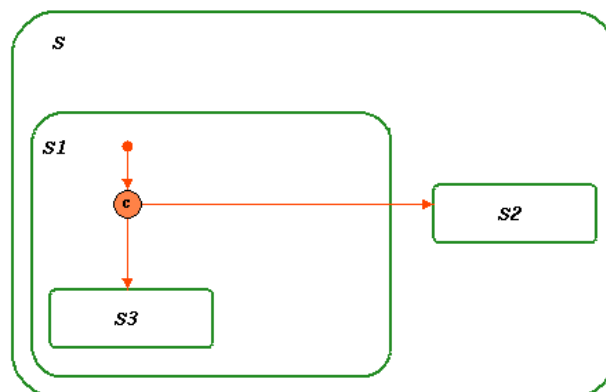


**Erroneous Elements:**

Components of the erroneous transitions

# (C1037) Compound transitions with nonorthogonal sources

There are compound transitions whose source set of states contains nonorthogonal (exclusive-or) states.

### Error In:

Correctness

### Example:

In the following figure, s1 and s2 are nonorthogonal states whose exiting transitions are contained in an illegal compound transition.
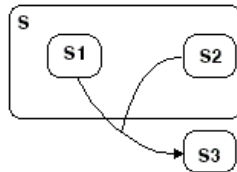


### Erroneous Elements:

Components of the erroneous transitions

# (C1039) Compound transitions with nonorthogonal targets

There are compound transitions whose target set of states contains nonorthogonal (exclusive-or) states.

**Error In:**

Correctness

**Example:**

In the following figure, the target states of `S1` contain the exclusive-or states `S2` and `S3`.
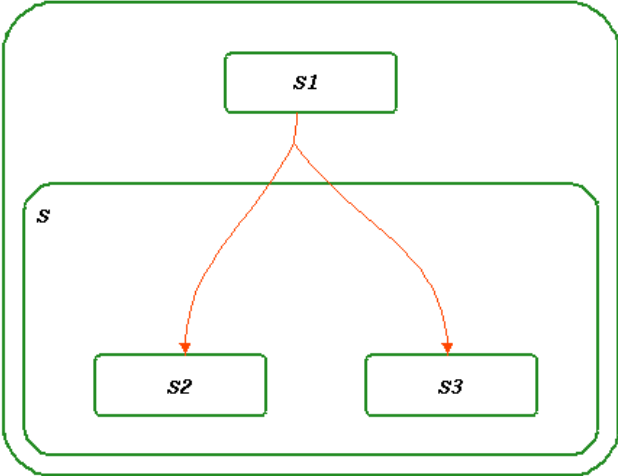


**Erroneous Elements:**

Components of the erroneous transitions

# (C1041) Or states without a default entrance

There are states of type "or" (decomposed into exclusive states) with entering transitions but with no "default" entrance.

### Error In:

Correctness

### Example:

In the following figure, s1 is an or state consisting of s2 and s3. A transition triggered by event E enters s1, but because s1 has no default entrance, the actual target of this transition is not defined.
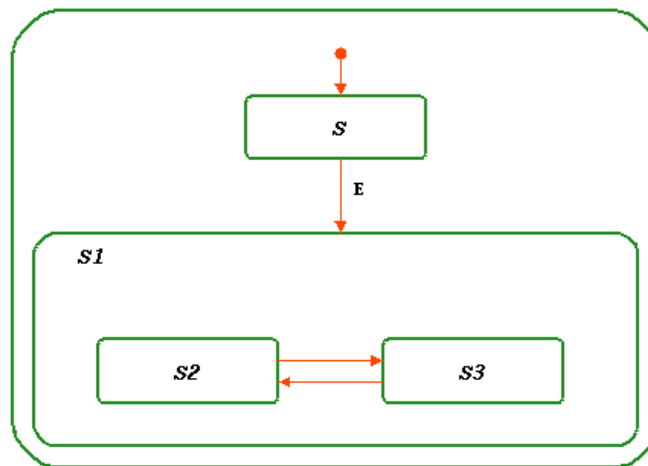
### Erroneous Elements:

States

# (C1043) The tested statechart has multiple top-level states and no default entrance

There are several top-level states in the statechart and no default transition is drawn to any of them.

**Error In:**

Correctness

**Erroneous Elements:**

The test chart

# (C1045) Instance off-page connectors with same name as in-page connector

Diagram connectors drawn in an instance state have the same name as another in-page connector in the chart.

**Error In:**

Correctness

**Example:**

In the following figure, the instance off-page connector `AB12` erroneously has the same name as the in-page connector, the target of the transition from `D`.
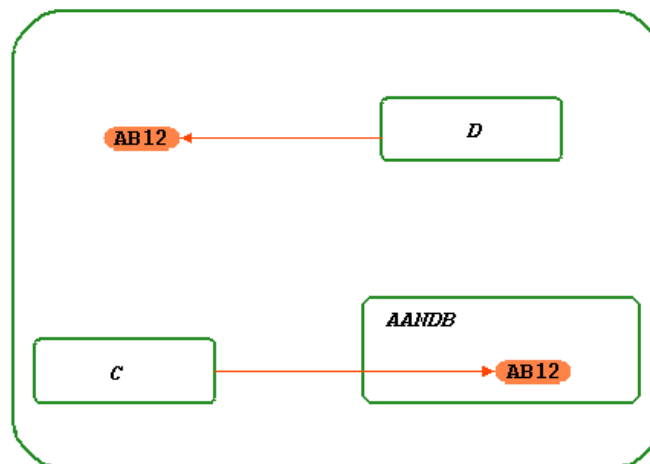
**Erroneous Elements:**

Instance connectors

# (C1047) Multiple instance off-page connectors with conflicting arrow directions

Diagram connectors are drawn in an instance state in which another connector having the same name but conflicting arrow directions is drawn.

**Error In:**

Correctness

**Erroneous Elements:**

Instance connectors

# (C1049) Definition off-page connectors with same name as in-page connector

Definition off-page connectors match to an instance off-page connector and have the same name as an in-page diagram connector.

**Error In:**

Correctness

**Example:**

In the following figure, the definition chart B has an off-page diagram connector AB12. The off-page connector has the same name as the in-page connector and is therefore illegal.
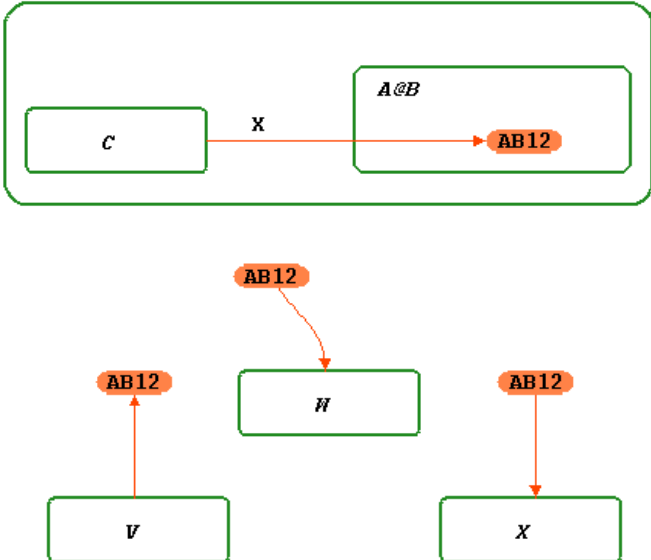


**Erroneous Elements:**

Connectors

# (C1051) Definition off-page connectors with conflicting arrow directions in instance off-page connector

There are definition off-page connectors whose matching connector in the instance chart has a conflicting arrow direction.

**Error In:**

Correctness

**Example:**

In the following figure, the instance A@B has an arrow entering the diagram connector AB12. Its definition chart B, however, has an arrow exiting the top-level box to the connector AB12.
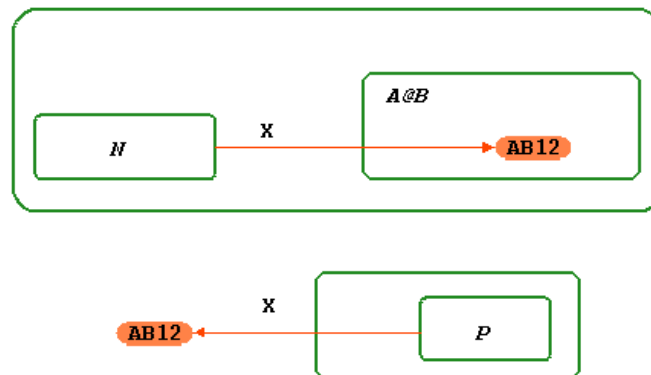


**Erroneous Elements:**

Definition off-page connectors

# (C1053) Generic instances with inconsistent parameter binding

There is inconsistency between the actual bindings in the instance box and the defined formal parameters in the generic chart.

**Error In:**

Correctness

**Erroneous Elements:**

Instance boxes

# (C1055) The tested chart has combinational assignments and it is instantiated by states

The tested chart contains combinational assignments and is used on an off-page chart for some state. This situation is illegal because it is misleading.

**Error In:**

Correctness

**Erroneous Elements:**

The tested chart

# (C1057) Combinational elements aliases of registered elements or vice versa

An alias data-item is used as a combinational element. It is assigned a value in a combinational assignment, whereas the element that displays in its definition is used as a registered element, or vice versa.

**Error In:**

Correctness

**Erroneous Elements:**

Data-items

# (C1059) Combinational elements assigned a value multiple times in the same entity

A data-item is assigned a value in two separate combinational assignments in the tested statechart.

**Error In:**

Correctness

**Erroneous Elements:**

Data-items

# (C1061) Elements using field references that do not appear in the record/union definition

Elements contain expressions in which a field of an explicitly defined record or union is used, but the field does not appear in the record or union definition.

**Error In:**

Correctness

**Erroneous elements:**

States, transitions, and textual elements

# (C1062) Elements erroneously using field references

Elements erroneously reference fields.

**Error In:**

Completeness

**Erroneous Elements:**

Any element referencing fields.

# (C1063) Formal parameters whose data-type is not global

Formal parameters are defined to be of some User-Defined Type (UDT), but the UDT is defined in the generic chart. The UDT is not visible outside the generic chart; therefore, an actual binding cannot have this type.

**Error In:**

Correctness

**Erroneous Elements:**

Data-items (formal parameters)

# (C1065) Actual binding whose data-type is not global

Data-items used as actual bindings in an instance box are defined to be of some User-defined Type (UDT), but the UDT is not global (it is defined in some chart, and not in a GDS). This UDT is not visible to the generic chart; therefore, a formal parameter cannot have this type.

**Error In:**

Correctness

**Erroneous Elements:**

Data-items

# (C1067) Empty Records/Unions

Data-items or data-types are defined to be a record or union, but there are no fields defined in the structure. This is a correctness error because it is a fatal error situation for some tools.

**Error In:**

Correctness

**Erroneous Elements:**

Data-items or user-defined types

# (C1071) Tasks/Procedures that are used incorrectly

Tasks can be bound only to activities. Any other usage (for example, in expressions) is incorrect. Procedures and functions cannot be bound to activities.

**Error In:**

Correctness

**Example:**

`TASK PRINTER` is used as procedure/function in the definition of `SCH:AN1`.

**Erroneous Elements:**

Subroutines

# (C1075) Elements containing out of range assignments of enumerated values

An element of some subtype is assigned a value that belongs to the enumerated type, but is not in the range of the subtype.

**Error In:**

Correctness

**Example:**

```
in definition of SCH_ADA:ACTION WORKING_DAY=SAT
```

**Erroneous Elements:**

Textual elements, boxes, and arrows

# (C1077) Procedures/Functions that are used incorrectly

Either procedures are used as functions (treated in some expression as if they return a value), or functions are used as procedures (called in expression where an action is expected).

**Error In:**

Correctness

**Example:**

Function FACTORIAL used as Procedure in definition of SCH_ADA:ACTION: FACTORIAL(Y)

**Erroneous Elements:**

Subroutines

# (C1079) Callback procedures that are used/defined incorrectly

Procedures are used as callbacks, whose first parameter is not of type string or not of mode Input.

**Error In:**

Correctness

**Example:**

```
CALLBACK P1, is an "Output" parameter
```

**Erroneous Elements:**

Subroutines

# (C1081) Elements containing subroutine calls with conflicting length of parameters

There are subroutine calls where the actual parameter's length conflicts with the corresponding formal parameter definition, or whose length is unknown.

**Error In:**

Correctness

**Example:**

```
STATE1
CALLBACK(A,ARRAY,3)
ARRAY - Unknown length
```

**Erroneous Elements:**

Textual elements, boxes, and arrows

# (C1083) Procedural statechart without termination connector

There is a procedural statechart with no termination connector. This means that any call to the connected subroutine will never end.

**Error In:**

Correctness

**Erroneous Elements:**

Charts

# (C1085) Elements not declared as parameters or as variables

There are elements used in the tested procedural statechart/flowchart that are not declared as local variables, parameters, or global data of the connected procedure.

**Error In:**

Correctness

**Example:**

```
data-item L2
condition C
```

**Erroneous Elements:**

Data-items and conditions

# (C1087) Local variables overriding parameters or global definitions

A local variable of the tested procedural statechart/flowchart overrides a parameter or global definition of the connected procedure.

**Error In:**

Correctness

**Example:**

```
P1 overriding parameter
```

**Erroneous Elements:**

Local variables

# (C1089) Subroutines using recursive calls

Subroutines use recursive calls (either directly or by other subroutines).

**Error In:**

Correctness

**Example:**

```
SR3 called from SR2
SR2 called from SR3
```

**Erroneous Elements:**

Subroutines

# (C1091) Subroutines using elements not declared as parameters, local-variables or globals

Subroutines that use elements are not declared as parameters, local-variables, or globals. All elements used in the subroutine code should be declared, including constant data-items and conditions, enumerated values, other subroutines, and user-defined types.

**Error In:**

Correctness

**Example:**

```
subroutine F L2
```

**Erroneous Elements:**

Subroutines

# (C1093) Subroutines that contain references to activities

The subroutine contains references to activities.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines

# (C1095) Subroutines with illegal parameter type or return type

There is a subroutine where a User-Defined Type (UDT) is the return type, or the type of a parameter. The following table below lists the illegal definitions for UDTs.

| Type | Return Type | Parameter |
|------|-------------|-----------|
| **UDT array of string** | (*) | **Illegal** |
| **UDT array of bit-array** | (*) | **Illegal** |
| **UDT array of direct R/U** | (*) | **Illegal** |
| **UDT queue of anything** | **Illegal** | **Illegal** |
| **Array of queue (any dimension)** | **Illegal** | **Illegal** |

(*) In these cases, a middle UDT is needed. For example, if the return type of a function should be an array of string, it should be defined as:

```
UDT array of STR_UDT, and STR_UDT, a string
```

**Error In:**

Correctness

**Format:**

```
subroutine <subroutine name>
return type <user-def type>, defined as <type>
    <user-def type>, type of parameter <parameter>,
    defined as <type>
```

**Example:**

```
subroutine F1
return type ARR_STR1, defined as array of string
    STR_QUEUE1, type of parameter P1, defined as queue
```

**Erroneous Elements:**

Subroutines

# (C1097) Subroutines with illegal local variables type

Local variables are defined as a User-Defined Type (UDT), as a queue, or an array (of any dimension) of a queue.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines

# (C1099) Local variables with illegal type

Local variables are defined as a user-defined type (UDT), as a queue, or an array (of any dimension) of a queue.

**Error In:**

Correctness

**Example:**

```
subroutine F1

STR_QUEUE1, type of local variable LV1, defined
        as queue
```

**Erroneous Elements:**

Subroutines

# (C1101) Subroutines with illegal/missing return command

There are functions without a RETURN command, procedures with a RETURN command, or a function with a return type that conflicts with the returned value's type.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines

# (C1103) Subroutines that call other subroutines whose global elements are missing from the calling subroutine definition

Global elements of a subroutine, SUB, must be declared as global elements of any other subroutine that calls SUB.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines

# (C1105) Subroutines with local variables/parameters overriding parameters/globals

There are subroutines with local variables that override parameters or globals, or there are parameters that override globals.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines

# (C1107) Subroutines with illegal combination of type and implementation

There are subroutines with an illegal combination of type and implementation.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines

# (C1109) Elements containing a truth table with illegal factorization structure

United input cells are not nested in a previous union; or, there are two or more rows that share the same input section, but have different output sections.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines and actions

# (C1111) Elements containing a truth table with illegal combination of inputs

Elements are described by truth tables using the same element twice as an input (via aliases). This check ensures that the same element is not used twice as a whole, but does not check overlapping of slices.

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines and actions

# (C1113) Elements containing a truth table with illegal combination of outputs

Elements are described by truth tables using the same element twice as an output (via aliases).

**Error In:**

Correctness

**Erroneous Elements:**

Subroutines and actions

# (C1115) Elements containing out of range integer/real assignments

Elements contain assignments of integers, where an out-of-range value is assigned to a restricted range integer, or one restricted range integer is assigned to another where there is no intersection of ranges. It tests if an integer or real value is compared with a value outside its range, as well as values assigned to integer or real variable.

**Error In:**

Correctness

**Erroneous Elements:**

States, actions, data-items, and transitions

# (C1117) Transitions without a return command whose target is a termination connector

If a procedural statechart implements a function and not a procedure, any transition that ends in a termination connector must return a value. Otherwise, the function is terminated without a return value.

**Error In:**

Correctness

**Erroneous Elements:**

Transitions

# (C1119) Transitions containing a return command whose target is not a termination connector

Functions end, but not in a termination connector.

**Error In:**

Correctness

**Erroneous Elements:**

Transitions

# (C1121) Transitions with incomplete source definition

There are transitions whose source is not connected to a state or connector. The source of such a transition displays in the statechart as a small line perpendicular to the transition.

**Error In:**

Completeness

**Erroneous Elements:**

Transitions

# (C1123) Transitions with incomplete target definition

There are transitions whose target is not connected to a state or connector. The target of such a transition displays in the statechart as a small line perpendicular to the transition.

**Error In:**

Completeness

**Erroneous Elements:**

Transitions

# (C1125) Switch/Case structure contains more than one default entry

The `switch_c` statement has more than one `default` case entry, or the `case_ada` statement has more than one `others when_ada` entry.

**Error In:**

Correctness

**Example:**

```
switch_c (X)
{
  case_c 1:
    Y=1;
  default:
    Y=0;
  case_c 2:
    Y=2;
  default:
    Y=3;
}
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C1127) Illegal switch structure: contains conditional break

The `break` action is not supported in `if` blocks.

**Error In:**

Correctness

**Example:**

```
switch(X) {
  case RED: y=5;
    if(COND1) then break
    endif;
  break
  .
  .
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C1129) More than one arrow with the 'else' trigger exit single source

There is more than one arrow with the 'else' trigger that are exiting from the same source.

**Error In:**

Correctness

**Erroneous Elements:**

Transitions, Flow Arrows

# (C1131) Illegal switch structure: non-unique selected entries

The `switch_c` structure contains a case entry that is repeated. Only the first entry will be executed.

**Error In:**

Correctness

**Example:**

```
switch_c (X)
{
  case_c 1:
    Y=1;
  case_c 1:
    Y=2;
  default:
    Y=3;
}
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C1133) Inconsistent initial value

An element has a default value that is inconsistent with its type and range definition.

**Error In:**

Correctness

**Example:**

Integer data-item X has a minimum value of 0, maximum value of 20, and a default value A. Integer data-item A is a constant set to 40. Because A is outside the minimum and maximum for X, it is an inconsistent initial value.

**Erroneous Elements:**

Integer and real data-items

# (C1135) Inconsistent range definition

This test warns about user-defined type definitions where the sub-type ranges are inconsistent with the parent type definition.

**Error In:**

Correctness

**Example:**

INT_TYPE is an integer User Defined Type and has a defined minimum/maximum range of 10 - 100 (as shown in the first figure below). SUB_INT_TYPE is a User Defined Type of INT_TYPE and has a range defined outside of 10 - 100 (as shown in the second figure below).
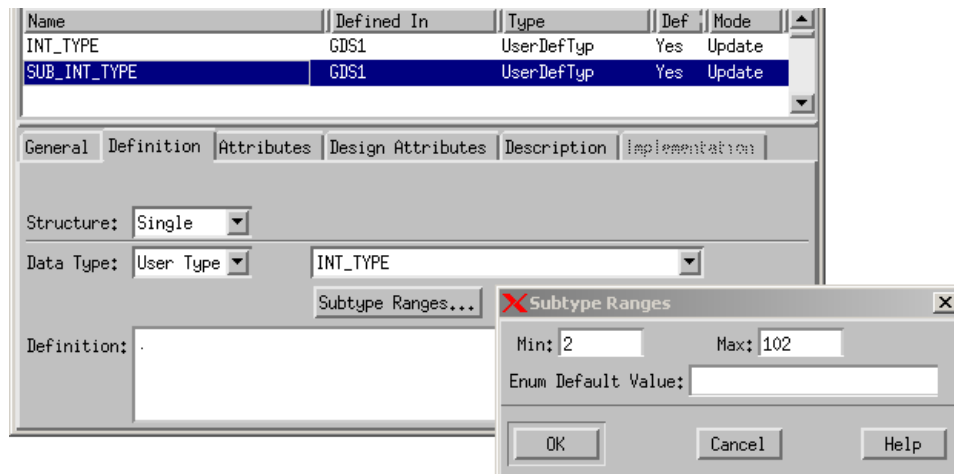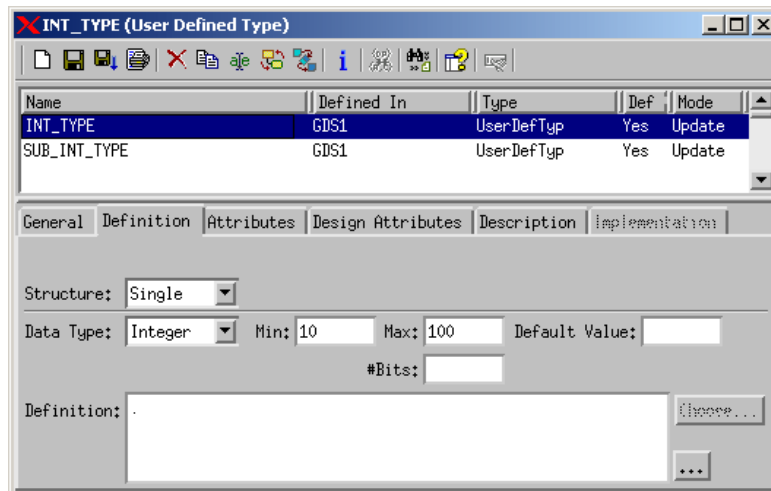
**Erroneous Elements:**

User-defined types

# (C1137) Empty information-flows

Array index expression does not result in an integer or contain elements with illegal type.

**Error In:**

Correctness

**Example:**

**Case I:**

Arr2 is an integer array from LeftInd2 to RightInd2.

LeftInd2 is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef2

SubDef2 is defined as:

Single None Constant with definition set to 2.34

**Case II:**

Arr1 is an integer array from LeftInd to RightInd.

LeftInd is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef

SubDef is defined as:

Single String Constant with definition set to 'My Definition'

# (C2001) Missing actual values

There is an instance box in whose property sheet there are formal parameters, without an actual binding.

**Error In:**

Completeness

**Erroneous Elements:**

Instance boxes

# (C2003) Elements with incorrect or incomplete attribute definitions

There are elements in whose property sheet there is an attribute name without a corresponding attribute value, or vice-versa.

**Error In:**

Completeness

**Erroneous Elements:**

Elements of all types

# (C2004) Elements with non-unique attribute definitions

Element contains more than one attribute with the same name.

**Error In:**

Completeness

**Erroneous Elements:**

Any element with attributes

# (C2005) Elements that are not referenced in Statemate expressions and definitions

There are textual elements explicitly defined in the properties, but there is no reference to them in the model.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements

# (C2007) Elements using redundant context variables

Elements contain actions with context variables that are assigned a value, but not used in the action.

**Error In:**

Completeness

**Erroneous Elements:**

States, transitions, and actions

# (C2009) Elements that could be defined lower in the hierarchy of charts

There are textual elements explicitly defined in the tested chart that are used only in other charts, which could be defined in the lower chart hierarchy without changing the resolution results.

The lowest chart in which they could be defined is the lowest common ancestor of all the charts in which references to this element exist.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements

# (C2011) Elements that reference unnamed boxes

Elements contain references to states that have no names. This situation can occur if a name of a referenced state is deleted.

**Error In:**

Completeness

**Erroneous Elements:**

Events, conditions, actions, states, and transitions

# (C2013) Elements containing references to components of non-array elements

**Error In:**

Completeness

**Description:**

Elements contain data-item expressions with indices, but the accessed data-item is not an array.

**Erroneous Elements:**

Actions, states, conditions, and transitions

# (C2015) Elements containing comparisons of unions: (not supported in Code Generation)

Elements contain comparisons of two union data-items. The comparisons of two unions is ambiguous because the result depends on which field of the unions is the current field in each element.

**Error In:**

Completeness

**Erroneous Elements:**

Actions, states, conditions, and transitions

# (C2021) Unlabeled transitions exiting from selection or condition connectors

There are transition segments whose source is either a selection connector or a condition connector, and no label is attached to them.

**Error In:**

Completeness

**Erroneous Elements:**

Transitions

# (C2023) Compound transitions without a trigger

There are compound transitions, none of whose segments contains a trigger part in their labels.

**Error In:**

Completeness

**Example:**

The following figure shows two compound transitions, where the second transition is in correct. The transition from S1 to S2 contains the trigger event E. The transition from S1 to S3 is labeled with an action A, but the trigger is missing.
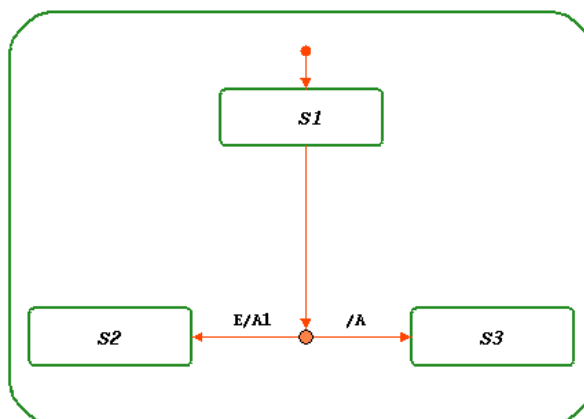
**Erroneous Elements:**

Components of the erroneous transitions

# (C2025) States without incoming transitions

There are states with no entering transition. Therefore, they are unreachable.

**Error In:**

Completeness

**Erroneous Elements:**

States

# (C2027) Basic states with a history connector

There are states that have no descendants, but have a history or deep history connector.

**Error In:**

Completeness

**Erroneous Elements:**

States

# (C2029) Basic states that are "history-cleared"

There are states that have no descendants, but are referenced in a clear history (`hc!`) action.

**Error In:**

Completeness

**Erroneous Elements:**

States

# (C2031) States with incomplete static reactions

There are states in which there is a static reaction that contains either a trigger or action, but not both.

**Error In:**

Completeness

**Erroneous Elements:**

States

# (C2033) States with static reaction triggered by condition

There are states with a static reaction whose trigger contains a condition without a preceding event (for example, `[C]`). When the condition is true, it is possible that a stationary situation is never reached. The static reaction is performed repeatedly, unless the condition changes its value or an event occurs that causes an exit from the state.

**Error In:**

Completeness

**Erroneous Elements:**

States

# (C2035) Isolated connectors

There are connectors with no entrances and no exits.

**Error In:**

Completeness

**Erroneous Elements:**

Connectors

# (C2037) Junction, selection, condition, history or diagram connectors with exits but no entrance

There are junction, selection, condition, or history connectors, or a group of in-page diagram connectors with the same name, having exits but no entrance. Therefore, there is no compound transition between states, including the existing transitions.

**Error In:**

Completeness

**Erroneous Elements:**

Connectors

# (C2039) Junction, selection, condition, or diagram connectors with entrances but no exit

There are junction, selection, or condition connectors or a group of in-page diagram connectors with the same name, having entrances but no exit. Therefore, there is no compound transition between states, including the entering transitions.

**Error In:**

Completeness

**Erroneous Elements:**

Connectors

# (C2041) Junction connectors with a single entrance and a single exit

There are junction connectors with a single entrance and a single exit. Therefore, they are probably redundant.

**Error In:**

Completeness

**Erroneous Elements:**

Connectors

# (C2043) Selection and condition connectors with entrances but only a single exit

There are selection and condition connectors with entrances, but only a single exit. Therefore, they are probably redundant.

**Error In:**

Completeness

**Erroneous Elements:**

Connectors

# (C2045) Unresolved elements

The referenced element is not defined.

**Error In:**

Completeness

**Erroneous Elements:**

All types of elements

# (C2047) The tested chart is a definition chart with multiple top-level boxes

The tested chart is used as an off-page definition of an instance state and has multiple top-level states.

**Error In:**

Completeness

**Example:**

In the following figure, the off-page chart B erroneously has more than one top level box (C and D).
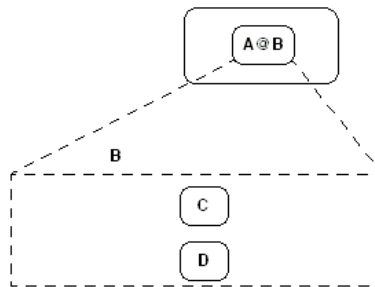


**Erroneous Elements:**

The tested chart

# (C2049) The tested chart is a definition chart with empty top-level box

The tested chart is used as an off-page definition of an instance state and has a single, empty, top-level state with no static reactions.

**Error In:**

Completeness

**Erroneous Elements:**

The tested chart

# (C2051) Instance off-page connectors without a matching definition off-page connector

Instance off-page connectors have no matching definition diagram connector.

**Error In:**

Completeness

**Erroneous Elements:**

The tested chart

# (C2053) "In" parameters not used in the chart

Input or InOut parameters are not used in the statechart.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C2055) "Out" parameters not affected (generated or modified) in the chart

Output or InOut parameters are not generated or modified in the statechart.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C2057) Elements that are used in the chart are neither "in" parameter nor affected (generated or modified) internally

Elements are used in a statechart but are neither input parameters nor affected internally.

**Error In:**

Completeness

**Erroneous Elements:**

Events, conditions, data-items, and activities

# (C2058) Elements affected (generated or modified) by the chart, but are neither "out" parameter nor used internally

Elements affected by the statechart are neither output parameters nor used internally.

**Error In:**

Completeness

**Erroneous Elements:**

Events, conditions, data-items, and activities

# (C2059) "In" parameters affected (generated or modified) in the chart

Input or InOut parameters are generated or modified in the statechart.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C2060) "Out" parameters used in the chart

Output or InOut parameters are used in the statechart.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C2061) Data elements with incomplete type definition

There are data elements whose type definition is missing or incomplete.

**Error In:**

Completeness

**Erroneous Elements:**

Events, conditions, data-items, and user-defined types

# (C2063) Constant parameter arrays with incomplete length definition

There are constant parameter arrays where either the definition of the formal parameter in the generic chart is incomplete, or they are bound to an actual parameter of undefined length.

**Error In:**

Completeness

**Erroneous Elements:**

Data-items and conditions

# (C2065) Defined elements that can be merged with elements from ancestor charts

Textual elements are explicitly defined in a chart that have matching, explicitly defined elements with the same name in ancestor charts. Both elements have the same type in both charts; therefore, they could probably be merged.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements

# (C2067) The tested statechart contains termination connector

The tested statechart contains a termination connector. This is improper in HW style, where all activities are always active.

**Error In:**

Completeness

**Erroneous Elements:**

The tested statechart

# (C2069) States with "throughout" or "within" activities

There are states with "throughout" or "within" activities. This is improper in HW style, where all activities are always active.

**Error In:**

Completeness

**Erroneous Elements:**

States

# (C2071) The tested Statechart contains actions, events or conditions related to activities

The tested statechart contains actions, events, or conditions related to activities. This is improper in HW style, where all activities are always active.

**Error In:**

Completeness

**Erroneous Elements:**

The tested statechart

# (C2073) The tested statecharts contains combinational assignments

The tested statechart contains combinational assignments.

**Error In:**

Completeness

**Erroneous Elements:**

The tested statechart

# (C2075) Elements overriding global definitions

There are textual elements with the same name as elements that are defined in a global definition set.

**Error In:**

Completeness

**Erroneous Elements:**

Textual elements

# (C2077) The tested statechart contains no states

The tested statechart contains no states.

**Error In:**

Completeness

**Erroneous Elements:**

The tested statechart

# (C2079) Subroutines not implemented in any language

There are subroutines that have no implementation in the Rational Statemate Action Language, Procedural Statechart/Flowchart, Ada Code, K&R C Code, or ANSI C Code.

**Error In:**

Completeness

**Example:**

```
CALLBACK
```

**Erroneous Elements:**

Subroutines

# (C2081) Elements containing triggers that use functions with potential side effects

There are functions used in the trigger part of labels, static- reactions, or mini-specs that have side-effects (they affect global data).

**Error In:**

Completeness

**Example:**

```
activity A1 Function F1 accessing global data,
    used in trigger: [Y==F1(X)]
```

**Erroneous Elements:**

States, activities, and transitions

# (C2083) Subroutines containing parameters defined as a nonunique User-defined type

A parameter of a subroutine is defined as a user-defined type, whose name is not unique in the model.

**Error In:**

Completeness

**Example:**

```
P2 parameter of procedure CALLBACK
Using nonunique user-defined type DAYS
```

**Erroneous Elements:**

Subroutines

# (C2085) Enumerated Types with nonunique value names

Enumerated types with values appear in other, unrelated enumerated type definitions (that is, not in a subtype).

**Error In:**

Completeness

**Example:**

```
user-defined type COLORS has a value of BLUE
UDT MOODS also has a value of BLUE
```

**Erroneous Elements:**

User-defined types

# (C2086) Data-Elements with Enumerated Value names

Textual element has same name as enumerated values.

**Error In:**

Correctness

**Erroneous Elements:**

Textual elements

# (C2087) Local variables that are not used

Local variables of a procedural statechart/flowchart are not used.

**Error In:**

Completeness

**Example:**

```
LOCAL1
```

**Erroneous Elements:**

Local variables

# (C2089) Local variables used but not affected (generated or modified)

Local variables are used, but not generated or modified anywhere (any such usage is uninitialized).

**Error In:**

Completeness

**Example:**

```
LOCAL1
```

**Erroneous Elements:**

Local variables

# (C2091) Local variables affected (generated or modified) but not used

Local variables are generated or modified, but are not used anywhere. These variables are redundant.

**Error In:**

Completeness

**Example:**

```
LOCAL1
```

**Erroneous Elements:**

Local variables

# (C2093) Textual subroutines using uninitialized local variables

Local variables of a textual procedure or function are used before they are assigned some value.

### Error In:

Completeness

### Example:

```
LOCAL1 in F1
```

### Erroneous Elements:

Subroutines

# (C2095) Textual subroutines using redundant local variables

Local variables of a textual procedure or function are generated or modified, but are not used.

### Error In:

Completeness

### Example:

```
LOCAL1 in F1
```

### Erroneous Elements:

Subroutines

# (C2097) Subroutines containing wrong usage of parameters or globals

In/Out/InOut parameters are not generated or modified, or are not used by the subroutine.

**Error In:**

Completeness

**Example:**

```
subroutine CALLBACK
Input parameter P1 not used
```

**Erroneous Elements:**

Subroutines

# (C2099) Subroutines with incomplete type definition

There are subroutines with incomplete type definitions.

**Error In:**

Completeness

**Erroneous Elements:**

Subroutines

# (C2101) Local variables not used in the subroutine

Local variables are not used in the subroutine code.

**Error In:**

Completeness

**Example:**

```
subroutine A
Local variable L3 not used
```

**Erroneous Elements:**

Subroutines

# (C2103) Elements with illegal "Selected Implementation"

The selected implementation is not one of the existing implementations of the element.

**Error In:**

Completeness

**Erroneous Elements:**

Subroutines, actions, and activities

# (C2105) Elements containing a truth table with empty default row

Elements described by truth tables contain a default row with the output columns blank and the action section empty.

**Error In:**

Completeness

**Erroneous Elements:**

Subroutines and actions

# (C2106) Elements containing a truth table with no inputs

Elements contain a truth table with no input column(s).

**Error In:**

Completeness

**Erroneous Elements:**

Elements containing truth tables

# (C2107) Elements containing a truth table with empty cells

Elements described by truth tables contain cells that are blank (cells that are not part of the default row).

**Error In:**

Completeness

**Erroneous Elements:**

Subroutines and actions

# (C2111) "when others" not last in case structure

Textual elements use themselves in their own definition, via a "chain" definition.

**Error In:**

Completeness

**Example:**

```
case_ada (X) is
    when_ada 1 =>
          Y=1;
    when_ada others =>
          Y=3;
    when_ada 2 =>
          Y=2;
end case_ada;
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C2113) Switch/Case structure missing default entry

The `switch_c` statement is missing the `default` case entry, or the `case_ada` statement is missing the `others when_ada` entry.

**Error In:**

Completeness

**Example:**

```
switch_c (X)
{
  case_c 1:
    Y=1;
  case_c 2:
    Y=2;
  }
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C15000) Events marked as 'Task Event' are not connected with EXTENDED Task

There are Events, which their "Its Task" Design-Attribute is not referring to a Task of type "Extended."

**Error In:**

Design

**Erroneous Elements:**

Events

# (C15001) Event(s), 'Its Task' is defined as 'default'

There are Events with their "Its Task" Design-Attribute set to "default."

**Error In:**

Design

**Erroneous Elements:**

Events

# (C15002) Either 'send' or 'receive' are operated on non-message elements

There are Data-Items, which are used with the operators send (sn!()) or receive (rc!()), without being defined as Messages in their Design-Attributes.

**Error In:**

Design

**Erroneous Elements:**

Data-Items

# (C15003) Either 'get' or 'release' are operated on non-resource elements

There are Conditions, which are used with the operators get (gt!()) or release (rl!()), without being defined as Resources in their Design-Attributes.

**Error In:**

Design

**Erroneous Elements:**

Conditions

# (C15004) Data-items and/or conditions mapping to external symbol, which is not unique

There is more than one Element with the same value to their "Bind Symbol Name" Design-Attribute.

**Error In:**

Design

**Erroneous Elements:**

Data-Items and Conditions

# (C15005) Operands in expressions, defined neither as double-buffered nor as having buffered access mode

There are Elements that are used with the operators changed (ch()), became-true (tr()) or became-false (fs()), and are not defined as "Double Buffered" or "Buffered Access" in their Design-Attributes.

**Error In:**

Design

**Erroneous Elements:**

Data-Items and Conditions

# (C15006) Multiple events, that are 'Task Event', with the same 'Its Task' and the same 'Event Mask'

There is more than one Event that are defined as "Task Event" in their Design-Attributes, and have the same value in their "Event Mask" Design-Attribute.

**Error In:**

Design

**Erroneous Elements:**

Events

# (C15007) Events that their its task is not appropriate

There are Events, which their "Its Task" Design-Attribute is set to an invalid "Task" name.

**Error In:**

Design

**Erroneous Elements:**

Events

# (C15008) Derived events that their its task is not appropriate

There are Events that their "Its-Task" Design-Attribute value is referring to a Task that is not any ancestor Activity.

**Error In:**

Design

**Erroneous Elements:**

Data-Items and Conditions

# (C15009) Elements unnecessarily defined as double-buffered

There are Elements that defined as "Double Buffered" in their Design-Attributes unnecessarily.

**Error In:**

Design

**Example:**

Data-Item DI1 has its Design-Attribute "Double Buffered" set to "yes", but it is not used with any of the operators tr(), fs() or ch().

**Erroneous Elements:**

Data-Items and Conditions

# (C16000) Boxes contain 'Combinational assignment' — Usage of incompatible feature

A state or statechart contains a combinational assignment that is not supported by the MicroC Code Generator.

**Error In:**

Compatibility between MicroC and Rational Statemate

**Erroneous Elements:**

Statecharts or states

# (C16003) Functions implemented with incompatible language

A subroutine is implemented using a language such as Ada, which is not supported by the MicroC Code Generator.

**Error In:**

Compatibility between MicroC and Rational Statemate

**Erroneous Elements:**

Subroutines

# (C16007) Usage of incompatible operators/functions

A statechart is using a built-in operator or function that is not supported by the MicroC Code Generator.

**Error In:**

Compatibility between MicroC and Rational Statemate

**Erroneous Elements:**

Actions, conditions, data-items, states, subroutines, and transition labels

# (C16008) Usage of incompatible operators on a record data-item

A statechart is using a built-in operator on a data-item of type `record`, which is not supported by the MicroC Code Generator.

**Error In:**

Compatibility between MicroC and Rational Statemate

**Erroneous Elements:**

Actions, conditions, data-items, states, subroutines, and transition labels

# Flow Chart Checks

This section documents the Check Model checks for flow charts. The checks are listed in numerical order.

## (C17001) Missing actual values for parameters.

### Error In:

Completeness

### Description:

There is an instance box in whose property sheet there are formal parameters, without an actual binding.

### Erroneous Elements:

Instance boxes

## (C17003) Elements with incorrect or incomplete attribute definitions

### Error In:

Completeness

### Description:

There are elements in whose property sheet there is an attribute name without a corresponding attribute value, or vice-versa.

### Erroneous Elements:

Elements of all types

# (C17004) Elements with non-unique attribute definitions

**Error In:**

Completeness

**Description:**

Element contains more than one attribute with the same name.

**Erroneous Elements:**

Any element with attributes

# (C17005) Elements are not referenced in Statemate expressions and definitions

**Error In:**

Completeness

**Description:**

There are textual elements explicitly defined in the properties, but there is no reference to them in the model.

**Erroneous Elements:**

Textual elements

# (C17007)Elements using redundant context variables

**Error In:**

Completeness

**Description:**

Elements contain actions with context variables that are assigned a value, but not used in the action.

**Erroneous Elements:**

States, transitions, and actions

# (C17009) Elements that could be defined lower in the hierarchy of charts

**Error In:**

Completeness

**Description:**

There are textual elements explicitly defined in the tested chart that are used only in other charts, which could be defined in the lower chart hierarchy without changing the resolution results.

The lowest chart in which they could be defined is the lowest common ancestor of all the charts in which references to this element exist.

**Erroneous Elements:**

Textual elements

# (C17011)Elements that reference unnamed boxes

### Error In:

Completeness

### Description:

Elements contain references to states that have no names. This situation can occur if a name of a referenced state is deleted.

### Erroneous Elements:

Events, conditions, actions, states, and transitions

# (C17013) Elements containing references to components of non-array elements

### Error In:

Completeness

### Description:

Elements contain data-item expressions with indices, but the accessed data-item is not an array.

### Erroneous Elements:

Actions, states, conditions, and transitions

# (C17015) Elements containing comparisons of unions: (not supported in Code Generation)

**Error In:**

Completeness

**Description:**

Elements contain comparisons of two union data-items. The comparisons of two unions is ambiguous because the result depends on which field of the unions is the current field in each element.

**Erroneous Elements:**

Actions, states, conditions, and transitions

# (C17017) Transitions with incomplete source definition

**Error In:**

Completeness

**Description:**

There are transitions whose source is not connected to a state or connector. The source of such a transition displays in the statechart as a small line perpendicular to the transition.

**Erroneous Elements:**

Transitions

# (C17019) Transitions with incomplete target definition

**Error In:**

Completeness

**Description:**

There are transitions whose target is not connected to a state or connector. The target of such a transition displays in the statechart as a small line perpendicular to the transition.

**Erroneous Elements:**

Transitions

# (C17025) States without incoming transitions

**Error In:**

Completeness

**Description:**

There are states with no entering transition. Therefore, they are unreachable.

**Erroneous Elements:**

States

# (C17045) Unresolved elements

**Error In:**

Completeness

**Description:**

The referenced element is not defined.

**Erroneous Elements:**

All types of elements

# (C17049) The tested chart is a definition with empty top-level box

**Error In:**

Completeness

**Description:**

The tested chart is used as an off-page definition of an instance state and has a single, empty, top-level state with no static reactions.

**Erroneous Elements:**

The tested chart

# (C17051) Instance off-page connectors without a matching definition off-page connector

**Error In:**

Completeness

**Description:**

Instance off-page connectors without a matching definition off-page connector.

**Erroneous Elements:**

The tested chart

# (C17053) "In" parameters not used in the chart

**Error In:**

Completeness

**Description:**

Input or InOut parameters are not used in the statechart.

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C17055) "Out" parameters not affected (generated or modified) in the chart

**Error In:**

Completeness

**Description:**

Output or InOut parameters are not generated or modified in the statechart.

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C17057) Elements that are used in the chart but are neither "in" parameter not affected (generated or modified) internally

**Error In:**

Completeness

**Description:**

Elements that are used in the chart but are neither "in" parameter nor affected (generated or modified) internally.

**Erroneous Elements:**

Events, conditions, data-items, and activities

# (C17058) Elements affected (generated or modified) by the chart but are neither "out" parameter not used internally

### Error In:

Completeness

### Description:

Elements affected (generated or modified) by the chart but are neither "out" parameter nor used internally.

### Erroneous Elements:

Events, conditions, data-items, and activities

# (C17059) "In parameters affected (generated or modified) in the chart

### Error In:

Completeness

### Description:

Input or InOut parameters are generated or modified in the statechart.

### Erroneous Elements:

Textual elements and activities (parameters)

# (C17060) "Out" parameters used in the chart

**Error In:**

Completeness

**Description:**

Output or InOut parameters are used in the statechart.

**Erroneous Elements:**

Textual elements and activities (parameters)

# (C17061) data elements with incomplete type definition

**Error In:**

Completeness

**Description:**

There are data elements whose type definition is missing or incomplete.

**Erroneous Elements:**

Events, conditions, data-items, and user-defined types

# (C17063) Constant parameter arrays with incomplete length definition

**Error In:**

Completeness

**Description:**

There are constant parameter arrays where either the definition of the formal parameter in the generic chart is incomplete, or they are bound to an actual parameter of undefined length.

**Erroneous Elements:**

Data-items and conditions

# (C17065) Defined elements that can be merged with elements from ancestor charts

### Error In:

Completeness

### Description:

Textual elements are explicitly defined in a chart that have matching, explicitly defined elements with the same name in ancestor charts. Both elements have the same type in both charts; therefore, they could probably be merged.

### Erroneous Elements:

Textual elements

# (C17071) The tested statechart contains actions, events, or conditions related to activities

### Error In:

Completeness

### Description:

The tested statechart contains actions, events, or conditions related to activities. This is improper in HW style, where all activities are always active.

### Erroneous Elements:

The tested statechart

# (C17075) Elements overriding global definition

**Error In:**

Completeness

**Description:**

There are textual elements with the same name as elements that are defined in a global definition set.

**Erroneous Elements:**

Textual elements

# (C17077) The test flowchart contains no flow-boxes

**Error In:**

Completeness

**Description:**

The tested flowchart contains no flow-boxes.

**Erroneous Elements:**

The tested flow chart

# (C17079) Subroutines not implemented in any language

**Error In:**

Completeness

**Description:**

There are subroutines that have no implementation in the Rational Statemate Action Language, Procedural Statechart/Flowchart, Ada Code, K&R C Code, or ANSI C Code.

**Example:**

```
CALLBACK
```

**Erroneous Elements:**

Subroutines

# (C17081) Elements containing triggers that use functions with potential side affects

**Error In:**

Completeness

**Description:**

There are functions used in the trigger part of labels, static- reactions, or mini-specs that have side-effects (they affect global data).

**Example:**

```
activity A1 Function F1 accessing global data,
    used in trigger: [Y==F1(X)]
```

**Erroneous Elements:**

States, activities, and transitions

# (C17083) Subroutines containing parameters defined as a nonunique User-defined type

**Error In:**

Completeness

**Description:**

A parameter of a subroutine is defined as a user-defined type, whose name is not unique in the model.

**Example:**

```
P2 parameter of procedure CALLBACK
Using nonunique user-defined type DAYS
```

**Erroneous Elements:**

Subroutines

# (C17085) Enumerated Types with nonunique value names

**Error In:**

Completeness

**Description:**

Enumerated types with values appear in other, unrelated enumerated type definitions (that is, not in a subtype).

**Example:**

```
user-defined type COLORS has a value of BLUE
UDT MOODS also has a value of BLUE
```

**Erroneous Elements:**

User-defined type

# (C17086) Data-Elements with Enumerated Value names

### Error In:

Correctness

### Description:

Textual element has same name as enumerated values.

### Erroneous Elements:

Textual elements

# (C17087) Local variables that are not used

### Error In:

Completeness

### Description:

Local variables of a procedural statechart/flowchart are not used.

### Example:

```
LOCAL1
```

### Erroneous Elements:

Local variables

# (C17089) Local variables used but not affected (generate or modified)

**Error In:**

Completeness

**Description:**

Local variables are used, but not generated or modified anywhere (any such usage is uninitialized).

**Example:**

LOCAL1

**Erroneous Elements:**

Local variables

# (C17091) Local variables affected (generated or modified) but not used

**Error In:**

Completeness

**Description:**

Local variables are generated or modified, but are not used anywhere. These variables are redundant.

**Example:**

LOCAL1

**Erroneous Elements:**

Local variables

# (C17093) Textual subroutines using uninitialized local variables

**Error In:**

Completeness

**Description:**

Local variables of a textual procedure or function are used before they are assigned some value.

**Example:**

```
LOCAL1 in F1
```

**Erroneous Elements:**

Subroutines

# (C17095) Textual subroutines using redundant local variables

**Error In:**

Completeness

**Description:**

Local variables of a textual procedure or function are generated or modified, but are not used.

**Example:**

```
LOCAL1 in F1
```

**Erroneous Elements:**

Subroutines

# (C17097) Subroutines containing wrong usage of parameters or globals

### Error In:

Completeness

### Description:

In/Out/InOut parameters are not generated or modified, or are not used by the subroutine.

### Example:

```
subroutine CALLBACK
Input parameter P1 not used
```

### Erroneous Elements:

Subroutines

# (C17099) Subroutines with incomplete type definition

### Error In:

Completeness

### Description:

There are subroutines with incomplete type definitions.

### Erroneous Elements:

Subroutines

# (C17101) Local variables not used in the subroutine

**Error In:**

Completeness

**Description:**

Local variables are not used in the subroutine code.

**Example:**

```
subroutine A
Local variable L3 not used
```

**Erroneous Elements:**

Subroutines

# (C17103) Elements with illegal "Selected Implementation"

**Error In:**

Completeness

**Description:**

The selected implementation is not one of the existing implementations of the element.

**Erroneous Elements:**

Subroutines, actions, and activities

# (C17105) Elements containing a truth table with empty default

**Error In:**

Completeness

**Description:**

Elements described by truth tables contain a default row with the output columns blank and the action section empty.

**Erroneous Elements:**

Subroutines and actions

# (C17107) Elements containing a truth table with empty cells

**Error In:**

Completeness

**Description:**

Elements described by truth tables contain cells that are blank (cells that are not part of the default row).

**Erroneous Elements:**

Subroutines and actions

# (C17108) Elements containing a truth table with no inputs

**Error In:**

Completeness

**Description:**

Elements contain a truth table with no input column(s).

**Erroneous Elements:**

Elements containing truth tables

# (C17109) "when others" not last in case structure

**Error In:**

Completeness

**Description:**

Textual elements use themselves in their own definition, via a "chain" definition.

**Example:**

```
case_ada (X) is
    when_ada 1 =>
          Y=1;
    when_ada others =>
          Y=3;
    when_ada 2 =>
          Y=2;
end case_ada;
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C17111) Switch/case structure is missing default entry

**Error In:**

Completeness

**Description:**

The `switch_c` statement is missing the `default` case entry, or the `case_ada` statement is missing the `others` `when_ada` entry.

**Example:**

```
switch_c (X)
{
  case_c 1:
    Y=1;
  case_c 2:
    Y=2;
  }
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C18000) Compound boxes with no start arrow and not default transition

**Error In:**

Completeness

**Description:**

Compound boxes with no start arrow and not default transition.

**Erroneous Elements:**

# (C18001) Compound boxes with no end connector

**Error In:**

Correctness

**Description:**

Compound box does not contain an end connector.

**Example:**
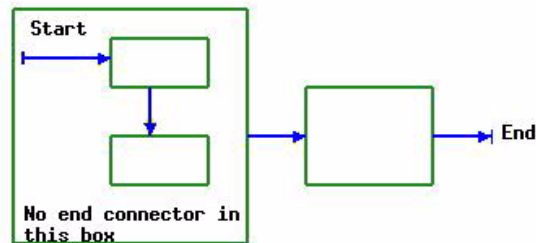


**Erroneous Elements:**

Flow chart compound box

# (C18002) Arrows crossing box boundaries

**Error In:**

Correctness

**Description:**

Arrow starts inside one box and ends in another box.

**Erroneous Elements:**

Arrows

# (C18003) Loops in element definition

**Error In:**

Correctness

**Description:**

Textual elements use themselves in their own definition, via a "chain" definition.

**Example:**

Compound event `E1` is defined as '`E2 or E3`'; `E2` is defined as '`E1 or E4`'.

**Erroneous Elements:**

All textual elements in the loop

# (C18004) Aliases defined as non variable or non bit-array

**Error In:**

Correctness

**Description:**

Aliases are illegally defined because the element appearing in the definition of the alias is defined as a non-variable compound or constant, or it is not a bit-array.

**Erroneous Elements:**

Data-items

# (C18005) Arrays with an out of range index

**Error In:**

Correctness

**Description:**

Arrays or bit-arrays are used with an index value that is out of the defined index range.

**Example:**

A is an integer array from 1 to 100. The model contains the following action:

```
A(101)=1
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C18007) Arrays with indices in reverse order

**Error In:**

Correctness

**Description:**

There are arrays or bit-arrays for which there are references to slices whose borders are in a reverse order relative to their type definition.

**Example:**

A is a bit-array from 31 down to 0. The model contains the following action:

```
A(0..7)=3
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C18009) Constants in inconsistent definition

**Error In:**

Correctness

**Description:**

There are either constant data-items or conditions whose values do not match their type definition; constant strings, arrays, or integers whose length does not match their defined length; or a constant integer whose value is not in the defined value range.

**Erroneous Elements:**

Data-item or condition

# (C18010) Data elements defined as arrays of bit

**Error In:**

Correctness

**Description:**

Elements are defined as array of a User-Defined Type (UDT), which is defined as bit.

**Example:**

`DI1` defined as array of `BIT1`, which is defined as bit.

**Erroneous Elements:**

Data-items, UDTs, and fields

# (C18011) Elements using uninitialized context variables

**Error In:**

Correctness

**Description:**

Elements contain action expressions that use an uninitialized context variable.

**Example:**

There is a transition labeled with the following label:

```
E / Y=$X; $X=Z
```
$X is used before any value is assigned to it.

**Erroneous Elements:**

States, transitions, and actions

# (C18013) Elements containing infinite while loops

**Error In:**

Correctness

**Description:**

This check detects cases in which the `while` loop control expression does not contain a context variable and there is no `break` statement in the loop.

**Erroneous Elements:**

States, transitions, and actions

# (C18015) Elements containing for loops with erroneous range indices

**Error In:**

Correctness

**Description:**

Elements contain action expressions in which there are `for` loop statements with invalid ranges.

**Example:**

There is an action that contains the following `for` loop statements:

```
for $I in 10 to 1 loop
...
for $I in 1 down to 10 loop
...
```

**Erroneous Elements:**

States, transitions, and actions

# (C18017) Numeric expressions using bit-arrays longer than 32 bits

**Error In:**

Correctness

**Description:**

There are elements containing numeric expressions that use bit-arrays longer than 32 bits. There is an automatic conversion from a bit-array to a numeric value only for a bit-array that is less than or equal to 32 bits.

**Erroneous Elements:**

States, transitions, and actions

# (C18019) Expressions containing operation between arrays with unequal length

**Error In:**

Correctness

**Description:**

Elements contain assignments or comparisons between arrays or array slices of different lengths.

**Example:**

There is an action that contains the assignment A=B, where A is defined as an array from 1 to 3, and B is an array from 1 to 5.

**Erroneous Elements:**

States, transitions, and actions

# (C18021) Loops in index definition

**Error In:**

Correctness

**Description:**

Data elements use constants in their definition in a "circular" manner.

**Example:**

AR is an array from 1 to N, while N is defined as rindex (AR).

**Erroneous Elements:**

Data-items, conditions, events, and user-defined types

# (C18023) Defined data elements that are used incorrectly

### Error In:

Correctness

### Description:

Elements contain expressions in which data-items that are explicitly defined to be of some data-type are used in a way that does not match the type consistency and conversion rules.

### Example:

$I$ is defined as an integer; $S$ is defined as a string. There is an action that contains the following assignment:

```
I=S
```

### Erroneous Elements:

States, transitions, events, conditions, data-items, and actions

# (C18025) Data elements without type definition that are used inconsistently

**Error In:**

Correctness

**Description:**

There are unresolved or explicitly defined data-items with no type definition that are used in expressions in an inconsistent manner.

**Example:**

$D$ is an unresolved element (it is not explicitly defined); $I$ is an integer; $S$ is a string. $D$ appears in the following assignments:

```
D=I; D=S
```

**Erroneous Elements:**

Data-items, conditions, and user-defined types

# (C18027) Predefined Statemate functions that are used incorrectly

**Error In:**

Correctness

**Description:**

Predefined Rational Statemate functions are used with parameters of a wrong type, the wrong number of parameters, or the wrong return type.

**Erroneous Elements:**

Functions

# (C18029) Subroutines that are used incorrectly

### Error In:

Correctness

### Description:

User functions are used inconsistently in different expressions (for example, with different return type, different parameter type, or a different number of parameters).

#### Note

When working with MicroC components, this test also reports situations where a subroutine is called with an actual 'out' parameter that is bound to an external symbol with get/set APIs.

### Erroneous Elements:

Functions

# (C18031) Decision box(es) with same boolean value on both exiting arrows

### Error In:

Correctness

### Description:

A decision box has two exiting arrows with the same value.

### Example:

```
<---yes---/\---yes--->
        \/
```

### Erroneous Elements:

Decision box

# (C18033) Default transitions with illegal sources

**Error In:**

Correctness

**Description:**

Transitions have multiple sources and one of them is either a conditional connector or a history connector.

**Example:**

In the following figure, the compound transition entering s4 (whose source set consists of the default connector in s1 and the state s2) is an erroneous transition.
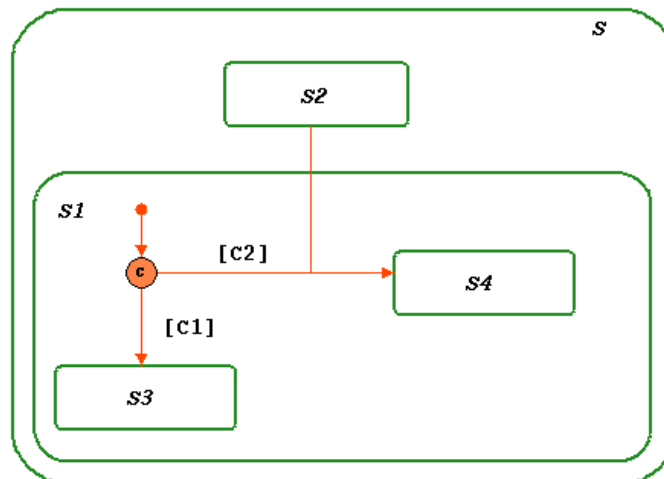


**Erroneous Elements:**

Components of the erroneous transitions

# (C18043) The tested statechart has multiple top-level states and no default entrance

**Error In:**

Correctness

**Description:**

There are several top-level states in the statechart and no default transition is drawn to any of them.

**Erroneous Elements:**

The test chart

# (C18045) Instance off-page connectors with same name as in-page connector

**Error In:**

Correctness

**Description:**

Diagram connectors drawn in an instance state have the same name as another in-page connector in the chart.

**Example:**

In the following figure, the instance off-page connector AB12 erroneously has the same name as the in-page connector, the target of the transition from D.
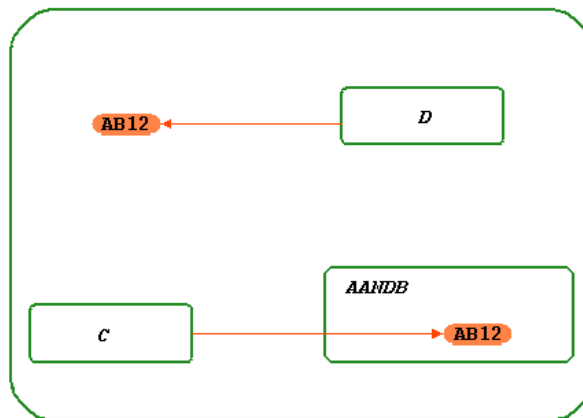
**Erroneous Elements:**

Instance connectors

# (C18047) Multiple instance off-page connectors with conflicting arrow directions

**Error In:**

Correctness

**Description:**

Diagram connectors are drawn in an instance state in which another connector having the same name but conflicting arrow directions is drawn.

**Erroneous Elements:**

Instance connectors

# (C18049) Definition off-page connectors with same name as in-page connectors

**Error In:**

Correctness

**Description:**

Definition off-page connectors match to an instance off-page connector and have the same name as an in-page diagram connector.

**Example:**

In the following figure, the definition chart B has an off-page diagram connector AB12. The off-page connector has the same name as the in-page connector and is therefore illegal.
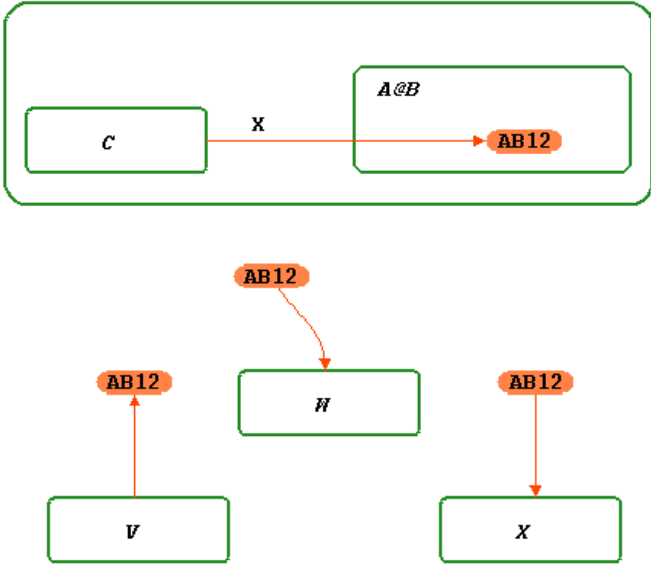


**Erroneous Elements:**

Connectors

# (C18051) Definition off-page connectors with conflicting arrow directions in instance off-page connector

**Error In:**

Correctness

**Description:**

There are definition off-page connectors whose matching connector in the instance chart has a conflicting arrow direction.

**Example**
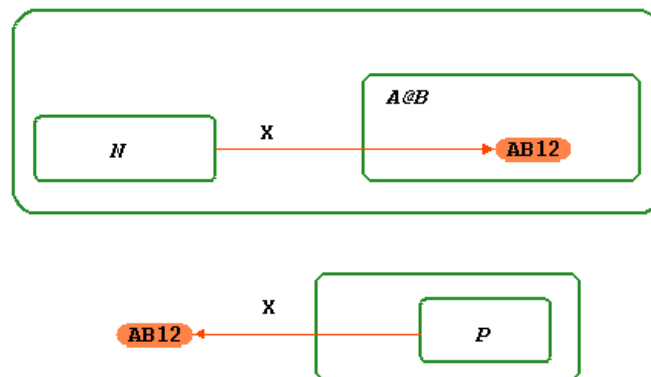


**Erroneous Elements:**

Definition off-page connectors

# (C18053) Generic instances with inconsistent parameter binding

**Error In:**

Correctness

**Description:**

There is inconsistency between the actual bindings in the instance box and the defined formal parameters in the generic chart.

**Erroneous Elements:**

Instance boxes

# (C18057) Combinational elements aliases of registered elements or vice versa

**Error In:**

Correctness

**Description:**

An alias data-item is used as a combinational element. It is assigned a value in a combinational assignment, whereas the element that appears in its definition is used as a registered element, or vice versa.

**Erroneous Elements:**

Data-items

# (C18061) Elements using field references that do not appear in the record/union definition

### Error In:

Correctness

### Description:

Elements contain expressions in which a field of an explicitly defined record or union is used, but the field does not appear in the record or union definition.

### Erroneous Elements:

States, transitions, and textual elements

# (C18062) Elements erroneously using field references

### Error In:

Completeness

### Description:

Elements erroneously reference fields.

### Erroneous Elements:

Any element referencing fields.

# (C18063) Formal parameters whose data-type is not global

**Error In:**

Correctness

**Description:**

Formal parameters are defined to be of some User-Defined Type (UDT), but the UDT is defined in the generic chart. The UDT is not visible outside the generic chart; therefore, an actual binding cannot have this type.

**Erroneous Elements:**

Data-items (formal parameters)

# (C18065) Actual binding whose data-type is not global

**Error In:**

Correctness

**Description:**

Data-items used as actual bindings in an instance box are defined to be of some User-Defined Type (UDT), but the UDT is not global (it is defined in some chart, and not in a GDS). This UDT is not visible to the generic chart; therefore, a formal parameter cannot have this type.

**Erroneous Elements:**

Data-items

# (C18067) Empty Records/Unions

### Error In:

Correctness

### Description:

Data-items or data-types are defined to be a record or union, but there are no fields defined in the structure. This is a correctness error because it is a fatal error situation for some tools.

### Erroneous Elements:

Data-items or user-defined types

# (C18075) Elements containing out of range assignments of enumerated values

### Error In:

Correctness

### Description:

An element of some subtype is assigned a value that belongs to the enumerated type, but is not in the range of the subtype.

### Example:

```
in definition of SCH_ADA:ACTION WORKING_DAY=SAT
```

### Erroneous Elements:

Textual elements, boxes, and arrows

# (C18077) procedures/Functions that are used incorrectly

**Error In:**

Correctness

**Description:**

Either procedures are used as functions (treated in some expression as if they return a value), or functions are used as procedures (called in expression where an action is expected).

**Example:**

Function FACTORIAL used as Procedure in definition of SCH_ADA:ACTION: FACTORIAL(Y)

**Erroneous Elements:**

Subroutines

# (C18079) Callback procedures that are used/defined incorrectly

**Error In:**

Correctness

**Description:**

Procedures are used as callbacks, whose first parameter is not of type string or not of mode Input.

**Example:**

```
CALLBACK P1, is an "Output" parameter
```

**Erroneous Elements:**

Subroutines

# (C18081) Elements containing subroutine calls with conflicting length of parameters

**Error In:**

Correctness

**Description:**

There are subroutine calls where the actual parameter's length conflicts with the corresponding formal parameter definition, or whose length is unknown.

**Example:**

```
STATE1
CALLBACK(A,ARRAY,3)
ARRAY - Unknown length
```

**Erroneous Elements:**

Textual elements, boxes, and arrows

# (C18085) Elements not declared as parameters or as variables

**Error In:**

Correctness

**Description:**

There are elements used in the tested procedural statechart/flowchart that are not declared as local variables, parameters, or global data of the connected procedure.

**Example:**

```
data-item L2
condition C
```

**Erroneous Elements:**

Data-items and conditions

# (C18087) Local variables overriding parameters or global definitions

**Error In:**

Correctness

**Description:**

A local variable of the tested procedural statechart/flowchart overrides a parameter or global definition of the connected procedure.

**Example:**

```
P1 overriding parameter
```

**Erroneous Elements:**

Local variables

# (C18089) Subroutine using recursive calls

**Error In:**

Correctness

**Description:**

Subroutines use recursive calls (either directly or by other subroutines).

**Example:**

```
SR3 called from SR2
SR2 called from SR3
```

**Erroneous Elements:**

Subroutines

# (C18091) Subroutines using elements not declared as parameters, local-variables, or globals

**Error In:**

Correctness

**Description:**

Subroutines that use elements are not declared as parameters, local-variables, or globals. All elements used in the subroutine code should be declared, including constant data-items and conditions, enumerated values, other subroutines, and user-defined types.

**Example:**

```
subroutine F L2
```

**Erroneous Elements:**

Subroutines

# (C18093) Subroutines that contain references to activities

**Error In:**

Correctness

**Description:**

The subroutine contains references to activities.

**Example:**

```
subroutine F
Activity A used in GLOBAL1
```

**Erroneous Elements:**

Subroutines

# (C18095) Subroutines with illegal parameter type or return type

**Error In:**

Correctness

**Description:**

There is a subroutine where a User-Defined Type (UDT) is the return type, or the type of a parameter. The following table below lists the illegal definitions for UDTs.

| Type | Return Type | Parameter |
|---|---|---|
| **UDT array of string** | (*) | **Illegal** |
| **UDT array of bit-array** | (*) | **Illegal** |
| **UDT array of direct R/U** | (*) | **Illegal** |
| **UDT queue of anything** | **Illegal** | **Illegal** |
| **Array of queue (any dimension)** | **Illegal** | **Illegal** |

(*) In these cases, a middle UDT is needed. For example, if the return type of a function should be an array of string, it should be defined as:

```
UDT array of STR_UDT, and STR_UDT, a string
```

**Format:**

```
subroutine <subroutine name>
return type <user-def type>, defined as <type>
    <user-def type>, type of parameter <parameter>,
    defined as <type>
```

**Example:**

```
subroutine F1
return type ARR_STR1, defined as array of string
    STR_QUEUE1, type of parameter P1, defined as queue
```

**Erroneous Elements:**

Subroutines

# (C18097) Subroutines with illegal local variable type

**Error In:**

Correctness

**Description:**

Local variables are defined as a User-Defined Type (UDT), as a queue, or an array (of any dimension) of a queue.

**Erroneous Elements:**

Subroutines

# (C18099) Local variables with illegal type

**Error In:**

Correctness

**Description:**

Local variables are defined as a user-defined type (UDT), as a queue, or an array (of any dimension) of a queue.

**Example:**

```
subroutine F1
STR_QUEUE1, type of local variable LV1, defined
     as queue
```

**Erroneous Elements:**

Subroutines

# (C18101) Subroutines with illegal/missing return command

**Error In:**

Correctness

**Description:**

There are functions without a RETURN command, procedures with a RETURN command, or a function with a return type that conflicts with the returned value's type.

**Erroneous Elements:**

Subroutines

# (C18103) Subroutines that call other subroutines whose global elements are missing from the calling subroutine definition

**Error In:**

Correctness

**Description:**

Global elements of a subroutine, SUB, must be declared as global elements of any other subroutine that calls SUB.

**Erroneous Elements:**

Subroutines

# (C18105) Subroutines with local variables/parameters overriding parameters/globals

**Error In:**

Correctness

**Description:**

There are subroutines with local variables that override parameters or globals, or there are parameters that override globals.

**Erroneous Elements:**

Subroutines

# (C18107) Subroutine with illegal combination of type and implementation

**Error In:**

Correctness

**Description:**

There are subroutines with an illegal combination of type and implementation.

**Erroneous Elements:**

Subroutines

# (C18109) Elements containing a truth table with illegal factorization structure

**Error In:**

Correctness

**Description:**

United input cells are not nested in a previous union; or, there are two or more rows that share the same input section, but have different output sections.

**Erroneous Elements:**

Subroutines and actions

# (C18111) Elements containing a truth table with illegal combination of inputs

**Error In:**

Correctness

**Description:**

Elements are described by truth tables using the same element twice as an input (via aliases). This check ensures that the same element is not used twice as a whole, but does not check overlapping of slices.

**Erroneous Elements:**

Subroutines and actions

# (C18113) Elements containing a truth table with illegal combination of outputs

**Error In:**

Correctness

**Description:**

Elements are described by truth tables using the same element twice as an output (via aliases).

**Erroneous Elements:**

Subroutines and actions

# (C18115) Elements containing out of range integer/real assignments

**Error In:**

Correctness

**Description:**

Elements contain assignments of integers, where an out-of-range value is assigned to a restricted range integer, or one restricted range integer is assigned to another where there is no intersection of ranges. It tests if an integer or real value is compared with a value outside its range, as well as values assigned to integer or real variable.

**Erroneous Elements:**

States, actions, data-items, and transitions

# (C18117) Switch/Case structure contains more than one default entry

**Error In:**

Correctness

**Description:**

The `switch_c` statement has more than one `default` case entry, or the `case_ada` statement has more than one `others when_ada` entry.

**Example:**

```
switch_c (X)
{
  case_c 1:
    Y=1;
  default:
    Y=0;
  case_c 2:
    Y=2;
  default:
    Y=3;
}
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C18119) Illegal switch structure: contains conditional break

**Error In:**

Correctness

**Description:**

The `break` action is not supported in `if` blocks.

**Example:**

```
switch(X) {
  case RED: y=5;
    if(COND1) then break
    endif;
  break
  .
  .
  .
```

**Erroneous Elements:**

Actions, states, and transition labels

# (C18121) Illegal switch structure: non-unique selected entries

### Error In:

Correctness

### Description:

The `switch_c` structure contains a case entry that is repeated. Only the first entry will be executed.

### Example:

```
switch_c (X)
{
  case_c 1:
    Y=1;
  case_c 1:
    Y=2;
  default:
    Y=3;
}
```

### Erroneous Elements:

Actions, states, and transition labels

# (C18123) Inconsistent initial value

### Error In:

Correctness

### Description:

An element has a default value that is inconsistent with its type and range definition.

### Example:

Integer data-item x has a minimum value of 0, maximum value of 20, and a default value A. Integer data-item A is a constant set to 40. Because A is outside the minimum and maximum for x, it is an inconsistent initial value.

### Erroneous Elements:

Integer and real data-items

# (C18125) Inconsistent range definition

**Error In:**

Correctness

**Description:**

This test warns about user-defined type definitions where the sub-type ranges are inconsistent with the parent type definition.

**Example:**

INT_TYPE is an integer User Defined Type and has a defined minimum/maximum range of 10 - 100 (as shown in the first figure below). SUB_INT_TYPE is a User Defined Type of INT_TYPE and has a range defined outside of 10 - 100 (as shown in the second figure below).
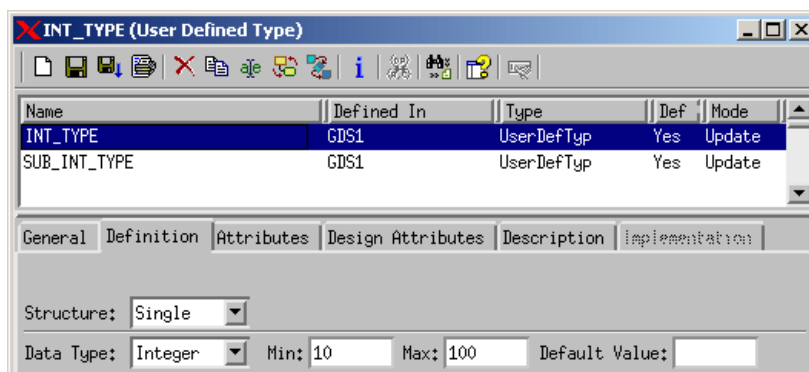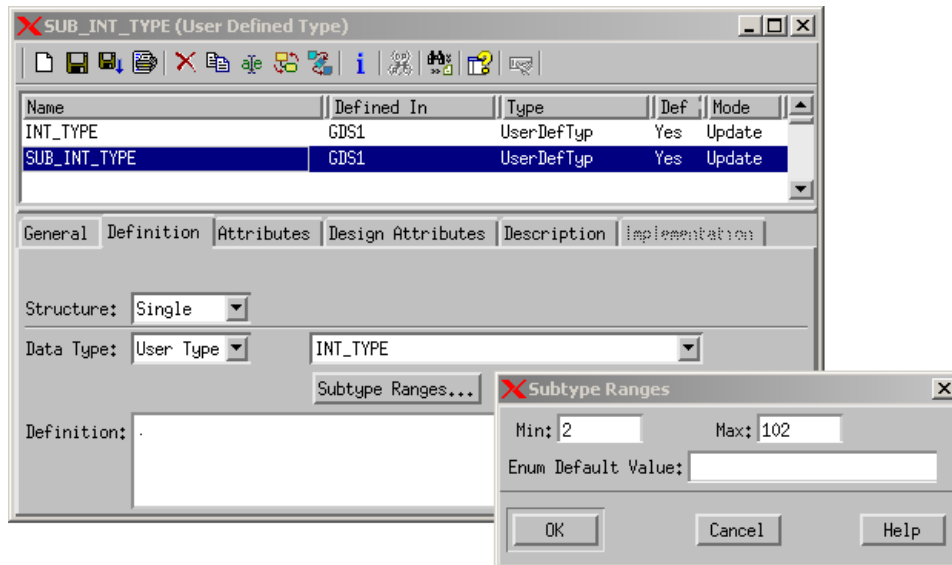
**Erroneous Elements:**

User-defined types

# (C18127) Empty information-flows

**Error In:**

Correctness

**Description:**

Array index expression does not result in an integer or contain elements with illegal type.

**Example:**

**Case I:**

Arr2 is an integer array from LeftInd2 to RightInd2.

LeftInd2 is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef2

SubDef2 is defined as:

**Case II:**

Arr1 is an integer array from LeftInd to RightInd.

LeftInd is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef

SubDef is defined as:

Single String Constant with definition set to 'My Definition'

# (C19000) Events(s), 'Its Task' is defined as 'default'

**Error In:**

Design

**Description:**

There are Events with their "Its Task" Design-Attribute set to "default".

**Erroneous Elements:**

Events

# (C19001) Events marked as 'Task Event' are not connected with ENTENDED Task.

**Error In:**

Design

**Description:**

There are Events, which their "Its Task" Design-Attribute is not referring to a Task of type "Extended".

**Erroneous Elements:**

Events

# (C19002) Either 'send' or receive' are operated on non-message elements

**Error In:**

Design

**Description:**

There are Data-Items, which are used with the operators send (sn!()) or receive (rc!()), without being defined as Messages in their Design-Attributes.

**Erroneous Elements:**

Data-Items

# (C19003) Either 'get' or 'release', are operated on non-resource elements

**Error In:**

Design

**Description:**

There are Conditions, which are used with the operators get (gt!()) or release (rl!()), without being defined as Resources in their Design-Attributes.

**Erroneous Elements:**

Conditions

# (C19004) Data-items or/and conditions, mapping to external symbol, which is not unique

**Error In:**

Design

**Description:**

There is more than one Element with the same value to their "Bind Symbol Name" Design-Attribute.

**Erroneous Elements:**

Data-Items and Conditions

# (C19005) Operands in expression, defined neither as double-buffered nor as having buffered access mode

**Error In:**

Design

**Description:**

There are Elements that are used with the operators changed (ch()), became-true (tr()) or became-false (fs()), and are not defined as "Double Buffered" or "Buffered Access" in their Design-Attributes.

**Erroneous Elements:**

Data-Items and Conditions

# (C19006) Multiple events, that are 'Task Event', with the same 'Its Task' and the same 'Event Mask'

**Error In:**

Design

**Description:**

There is more than one Event that are defined as "Task Event" in their Design-Attributes, and have the same value in their "Event Mask" Design-Attribute.

**Erroneous Elements:**

Events

# (C19007) Events that their its task is not appropriate

**Error In:**

Design

**Description:**

There are Events, which their "Its Task" Design-Attribute is set to an invalid "Task" name.

**Erroneous Elements:**

Events

# (C19008) Event(s), that are not 'Task Event", used in flowchart

**Error In:**

Design

**Description:**

Event element is defined in a flow chart but in not used as a "Task Event".

**Erroneous Elements:**

Event Element

# (C19009) Derived events that their its task is not appropriate

**Error In:**

Design

**Description:**

There are Events that their "Its-Task" Design-Attribute value is referring to a Task that is not any ancestor Activity.

**Erroneous Elements:**

Data-Items and Conditions

# (C19010) Elements unnecessarily defined as double-buffered

### Error In:

Design

### Description:

There are Elements that defined as "Double Buffered" in their Design-Attributes unnecessarily.

### Example:

Data-Item DI1 has its Design-Attribute "Double Buffered" set to "yes", but it is not used with any of the operators tr(), fs() or ch().

### Erroneous Elements:

Data-Items and Conditions

# (C20003) Usage of incompatible operators/functions

### Error In:

Compatibility between MicroC and Rational Statemate

### Description:

A statechart is using a built-in operator or function that is not supported by the MicroC Code Generator.

### Erroneous Elements:

Actions, conditions, data-items, states, subroutines, and transition labels

# (C20004) Usage of incompatible operators on a record data-item

**Error In:**

Compatibility between MicroC and Rational Statemate

**Description:**

A statechart is using a built-in operator on a data-item of type `record`, which is not supported by the MicroC Code Generator.

**Erroneous Elements:**

Actions, conditions, data-items, states, subroutines, and transition labels

# Activity Chart Checks

This section documents the Check Model checks for activity charts. The checks are listed in numerical order.

## (C3001) Loops in element definition

**Error In:**

Correctness

**Description:**

Textual elements use themselves in their definition via "chain" definitions.

**Example:**

Compound event `E1` is defined as '`E2` or `E3`'; `E2` is defined as '`E1` or `E4`.'

**Erroneous Elements:**

All textual elements in the loop

# (C3003) Aliases defined as non variable or non bit-array

**Error In:**

Correctness

**Description:**

Aliases, illegally defined because of the element appearing in the definition of the alias, are defined as non-variable (compound or constant) or non-bit-array.

**Example:**

x2 is a single bit-array from 1 to 14, defined in chart TOP. X2_INIT is a single integer, with usage of Alias, defined in chart TOP to be X2(2). The error occurs on data-item X2_INIT.

**Erroneous Elements:**

Data-items

# (C3005) Arrays with an out of range index

**Error In:**

Correctness

**Description:**

Arrays or bit-arrays are used with an index value out of the defined index range.

**Example:**

```
A is an integer array from 1 to 100. There is an action in the model:
A(101)=1
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C3007) Arrays with indices in reverse order

**Error In:**

Correctness

**Description:**

Arrays or bit-arrays reference slices whose borders are in reverse order relative to their type definition.

**Example:**

A is a bit-array from 31 down to 0. There is an action in the model:

```
A(0..7)=3
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C3009) Constants with inconsistent definition

**Error In:**

Correctness

**Description:**

One of the following situations:

- There are constant data-items or conditions whose values do not match their type definition.
- There are constant strings, arrays, or integers whose length does not match their defined length.
- There are constant integers whose values are not in the defined value range.

**Erroneous Elements:**

Data-items and conditions

# (C3010) Data elements defined as arrays of bit

**Error In:**

Correctness

**Description:**

There are elements defined as arrays of a user-defined type (UDT) that is defined as bit.

**Example:**

`DI1` defined as array of `BIT1`, which is defined as bit

**Erroneous Elements:**

Data-items, user-defined types, and fields

# (C3011) Elements using uninitialized context variables

**Error In:**

Correctness

**Description:**

Elements contain action expressions in which an uninitialized context variable is used.

**Example:**

A transition has the following label:

```
E/Y=$X; $X=Z
```

$X is used before any value is assigned to it.

**Erroneous Elements:**

Activities and actions

# (C3013) Elements containing infinite While Loops

**Error In:**

Correctness

**Description:**

Elements contain action expressions that have infinite `while` loop statements. The check detects cases where the `while` loop control expression does not contain a context variable or a `break` statement in the loop.

**Example:**

A mini-spec contains the following expression:

```
EV / while COND1 loop
A=B;
end loop
```

The expression contains no context variable or `break` statement.

**Erroneous Elements:**

Activities and actions

# (C3015) Elements containing For Loops with erroneous range indices

**Error In:**

Correctness

**Description:**

Elements contain action expressions that have `for` loops statements with an invalid range.

**Example:**

An action contains the following `for` loop statement:

```
for $I in 10 to 1 loop
```

The correct statement would use `downto`.

**Erroneous Elements:**

Activities and actions

# (C3017) Numeric expressions using bit-arrays longer than 32 bits

**Error In:**

Correctness

**Description:**

Elements contain numeric expressions that have bit-arrays longer than 32 bits. Conversion occurs automatically from a bit-array to a numeric value only for a bit-array with less than or equal to 32 bits.

**Erroneous Elements:**

Activities and actions

# (C3019) Expressions containing operations between arrays with unequal length

### Error In:

**Correctness**

### Description:

Elements contain assignments or comparisons between arrays or array slices of different lengths.

### Example:

An action contains the following assignment:

```
A=B
when A is defined as an array from 1 to 3,
and B is an array from 1 to 5.
```

### Erroneous Elements:

Activities, events, conditions, and actions

# (C3021) Loops in index definition

### Error In:

Correctness

### Description:

Data elements use constants in their definition in a "circular" manner.

### Example:

```
AR is an array from 1 to N
while N is defined as rindex (AR)
```

### Erroneous Elements:

Data-items, conditions, events, and user-defined types

# (C3023) Defined data elements that are used incorrectly

**Error In:**

Correctness

**Description:**

Elements contain expressions in which data-items that are explicitly defined to be of some data-type are used in a way that does not match the type consistency and conversion rules.

**Example:**

I is defined as an integer; S is defined as a string. There is an action that contains the following assignment:

I=S

**Erroneous Elements:**

Activities, events, conditions, data-items, and actions

# (C3025) Data elements without type definition that are used inconsistently

**Error In:**

Correctness

**Description:**

Unresolved or explicitly defined data-items with no type definition or specified type are used in expressions in an inconsistent manner.

**Example:**

D is an unresolved or not explicitly defined element. I is an integer; S is a string. D appears in the following assignments:

```
D=I; D=S
```

**Erroneous Elements:**

Data-items, conditions, and user-defined types

# (C3027) Predefined functions that are used incorrectly

**Error In:**

Correctness

**Description:**

Predefined functions are used with parameters of a wrong type, the wrong number of parameters, or an incorrect return type.

**Example:**

A mini-spec contains the following expression, where THETA is defined as a real and C is defined as a condition:

```
    EV /C = sin(THETA)
C should be a real.
```

**Erroneous Elements:**

Functions

# (C3029) User functions that are used inconsistently

**Error In:**

Correctness

**Description:**

User functions are used inconsistently in different expressions as in a different return type, parameter type, or number of parameters.

> **Note**
>
> When working with MicroC components, this test also reports situations where a subroutine is called with an actual 'out' parameter that is bound to an external symbol with get/set APIs.

**Example:**

A subroutine `SUB_1` is defined as a function with two input parameters that returns an integer. An activity `ACT_1` contains the following mini-spec, where `Y` is defined as an integer:

```
EV_1/Y=SUB_1(A1,B1)
```

A second activity on the same chart contains the following mini-spec, where `Z` is defined as a condition:

```
EV_2/Z=SUB_1(A2,B2)
```

The error is generated on the `SUB_1` function because the return type is used as an integer in the first activity and as a condition in the second activity.

A third activity on the same chart contains the following mini-spec:

```
EV_3/W=SUB_1(A3,B3,C3)
```

The error is also generated in this case due to an inconsistent number of parameters.

**Erroneous Elements:**

Functions

# (C3031) Loops in compound flow-lines

**Error In:**

Correctness

**Description:**

A group of flow-line segments, linked through connectors, create a loop or cycle that does not contain an activity or data-store.

**Example:**

In the following figure, the data-flow with label z is shown as having the same source and target (diagram connector AB), thus creating a loop.
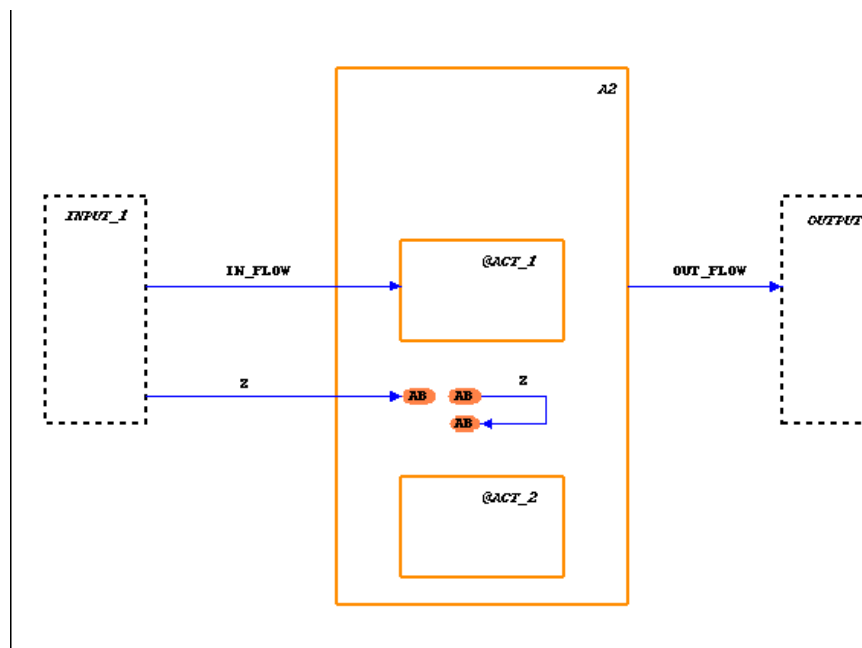


**Erroneous Elements:**

Flow-line segments in loops.

# (C3033) Compound flow-lines with identical sources and targets

**Error In:**

Correctness

**Description:**

Compound flow-lines with identical sources and targets (loopback flow-lines).

**Example:**

The following figure shows the simplest case:
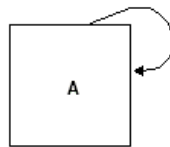


**Erroneous Elements:**

Compound flow-lines

# (C3035) Flow-line segments with illegal joints

**Error In:**

Correctness

**Description:**

Compound flow-lines are constructed by using joint connectors in a confusing order.

**Example:**
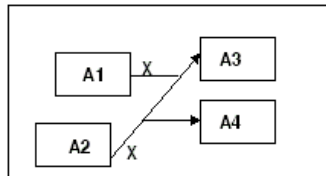


**Erroneous Elements:**

Components of the erroneous flow-lines

# (C3037) Flow-lines between data-stores

### Error In:

Correctness

### Description:

Compound flow-lines denote a flow of information between data-stores, which are passive elements.

### Example:

In the following figure, data-item A is shown flowing from STORE_1 to STORE_2 via a junction connector.
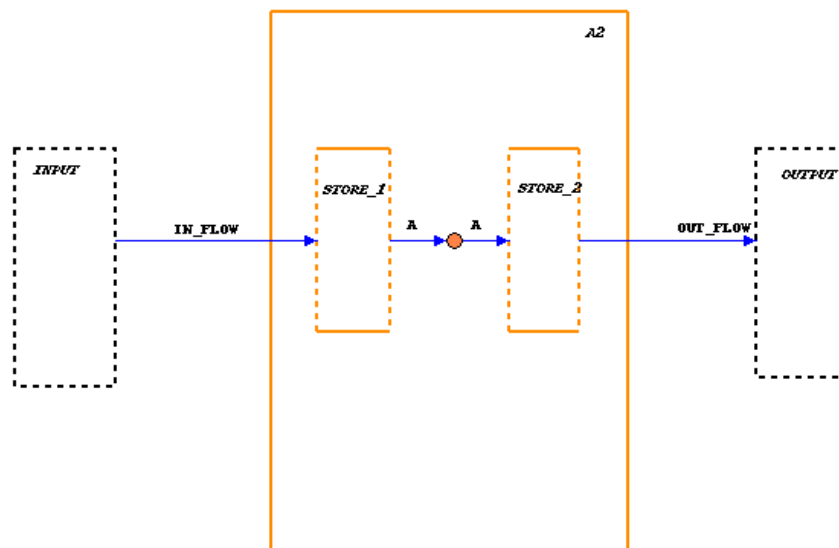
# (C3039) Flow-lines between external activities

**Error In:**

Correctness

**Description:**

There are compound flow-lines between external activities. These flow-lines, even if they exist, should not be specified in the chart where the source and target activities are external, and therefore out of the description scope.

**Example:**

In the following figure, data-item `A` is shown flowing between external activities `INPUT_1` and `INPUT_2`.
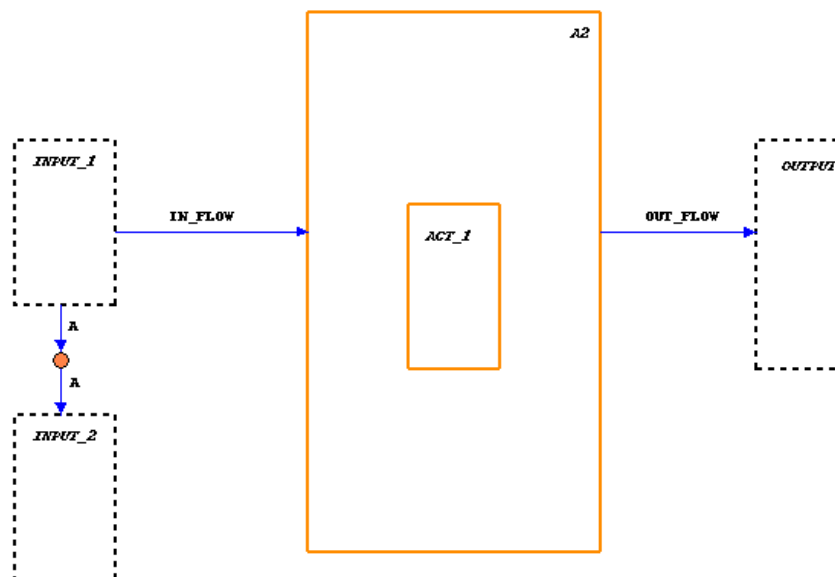


**Erroneous Elements:**

- ◆ Components of the erroneous flow-lines

# (C3041) Conflicting labels on flow-line components

**Error In:**

Correctness

**Description:**

There are compound flow-lines whose segments are labeled with conflicting labels.

**Example:**

In the following figure, if x and y include different data elements (such as events, conditions, or data-it ems), the flow-line from A1 and A2 to A3 is incorrect.
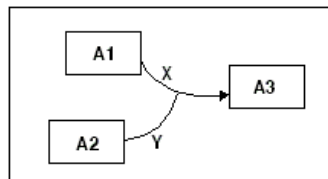


As shown in the following figure, the compound flow-line from A2 to A3 is incorrect because its labels are x and y. However, the compound flow-line from A1 to A3 is correct.
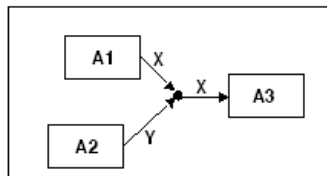


**Erroneous Elements:**

Components of the erroneous flow-lines

# (C3043) Composition connectors with multiple entrances and multiple exits

**Error In:**

Correctness

**Description:**

There are composition connectors with multiple entering flow-lines and multiple exiting flow-lines.

**Erroneous Elements:**

Connectors

# (C3045) Instance off-page connectors with same name as in-page connector

**Error In:**

Correctness

**Description:**

There are off-page diagram connectors drawn in an instance activity that have the same name as another in-page connector in the chart.

**Example:**

As shown on the following figure, the off-page diagram connector drawn in @ACT_1 has the same name as the in-page connector CON1.
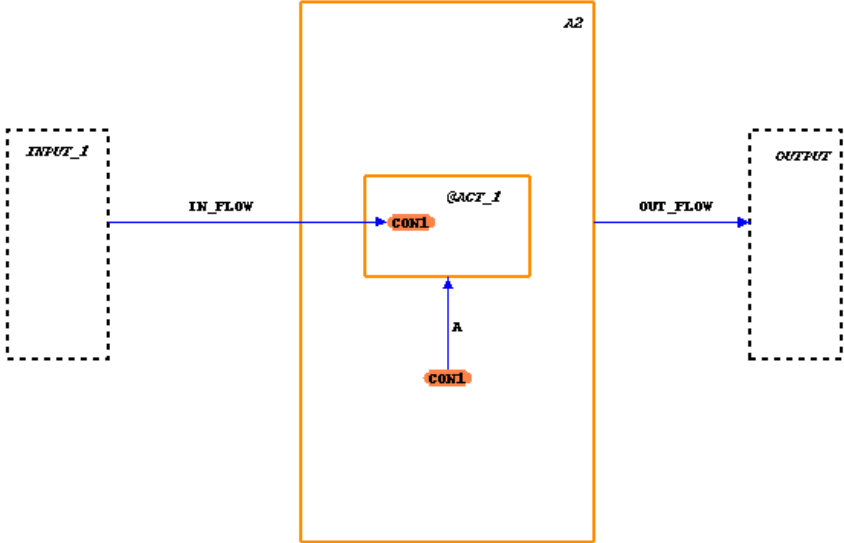
**Erroneous Elements:**

Instance connectors

# (C3047) Multiple instance off-page connectors with conflicting arrow directions

**Error In:**

Correctness

**Description:**

There are off-page diagram connectors drawn in an instance activity in which another connector with the same name, but conflicting arrow directions, is drawn.

**Example:**

The following figure shows an off-page diagram connector drawn in @ACT_1 with an incoming arrow, which conflicts with a second off-page diagram connector with an outgoing arrow.
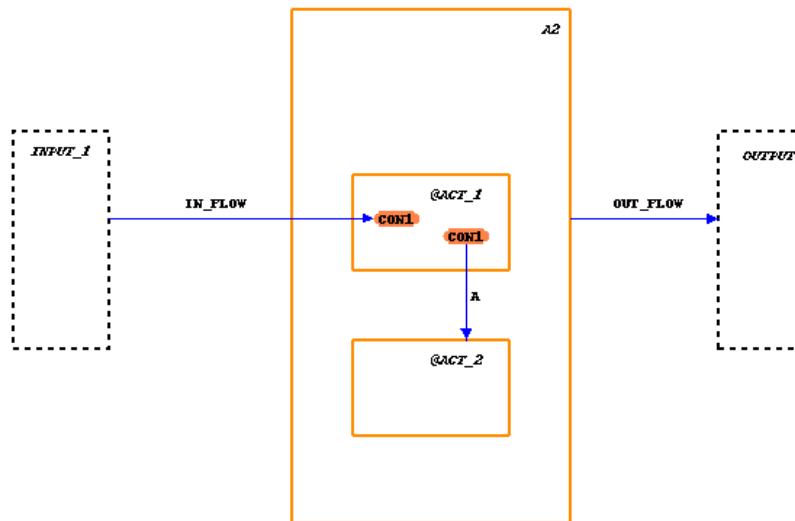


**Erroneous Elements:**

Instance connectors

# (C3049) Definition off-page connectors with same name as in-page connector

**Error In:**

Correctness

**Description:**

There is a definition off-page connector that matches an instance off-page connector and has the same name as an in-page diagram connector.

This check is performed only when you specify with definition charts. The error is reported in the definition chart.

**Erroneous Elements:**

Connectors

# (C3051) Definition off-page connectors with conflicting arrow directions in instance off-page connector

**Error In:**

Correctness

**Description:**

There are definition off-page connectors whose matching connector in the instance chart has a conflicting arrow direction.

This check is performed only when you specify with definition charts. The error is reported in the definition chart.

**Erroneous Elements:**

Definition off-page connectors

# (C3053) Elements that flow to/from instance with contradicting source/target name in definition chart

**Error In:**

Correctness

**Description:**

There are elements that flow to or from both an instance activity and an external activity in the off-page definition chart, but the source or target name in the off-page definition chart is not the same as the name or synonym in the instance chart.

This check is performed only when you specify with definition charts. The error is reported in the instance chart.

**Example:**

In the following figure, this error is generated when testing chart A2, where IN_FLOW flows from external activity INPUT_1 to instance activity ACT_1. However, in the off-page definition chart, IN_FLOW is shown flowing from external activity INPUT_2.
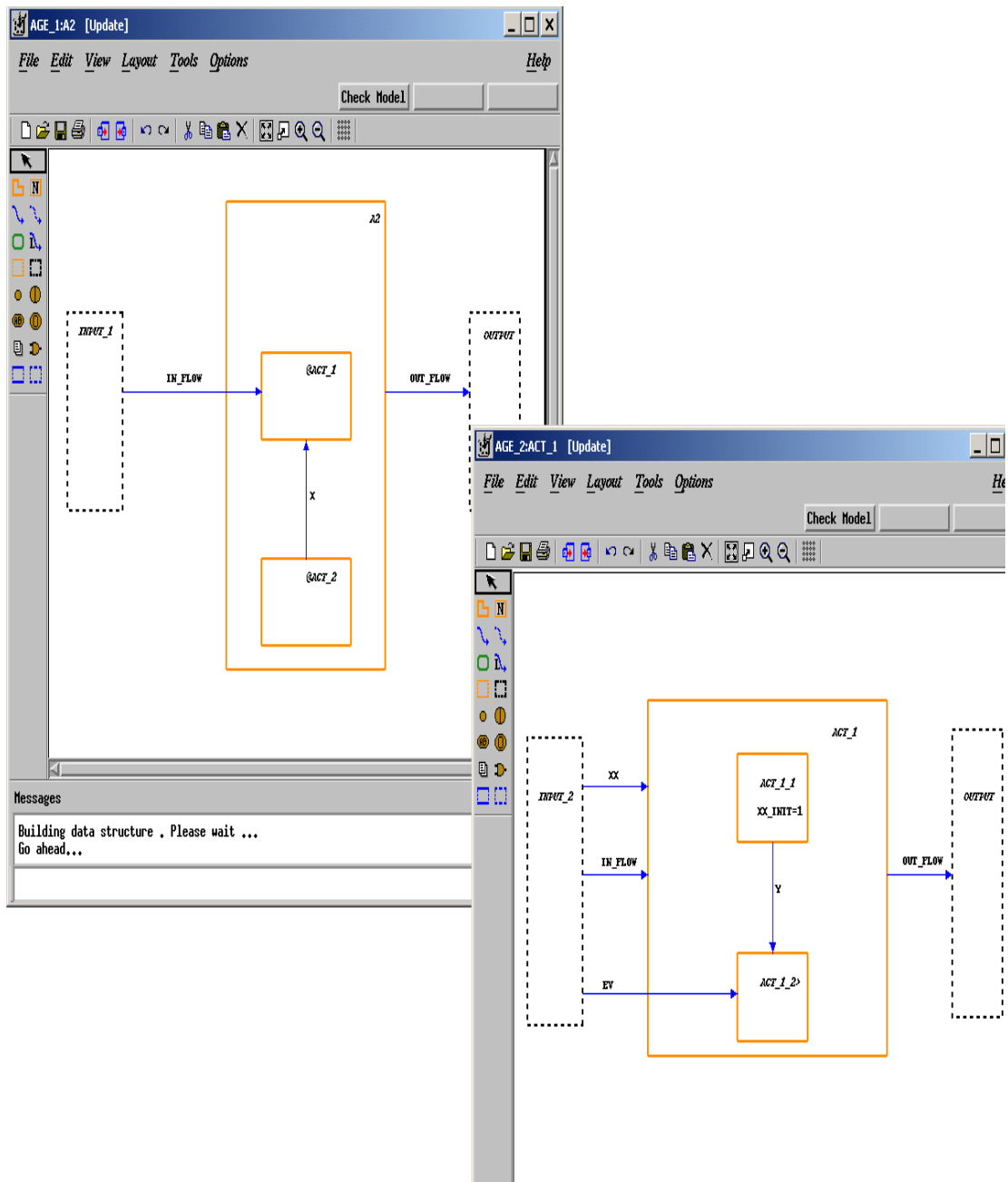
**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C3055) Elements that flow to/from definition with contradicting source/target name in instance chart

**Error In:**

Correctness

**Description:**

There are elements that flow to or from both an external activity in the off-page definition chart and the instance activity, but the source or target and the name or synonym in the instance are not the same as the name in the off-page definition chart. This check is performed only when you specify with definition charts. The error is reported in the definition chart.

**Example:**

In the following figure, this error is generated when testing chart AXT1, where IN_FLOW flows from external activity INPUT_2 to activity ACT_1. However, in the off-page definition chart, IN_FLOW is shown flowing from external activity INPUT_1.
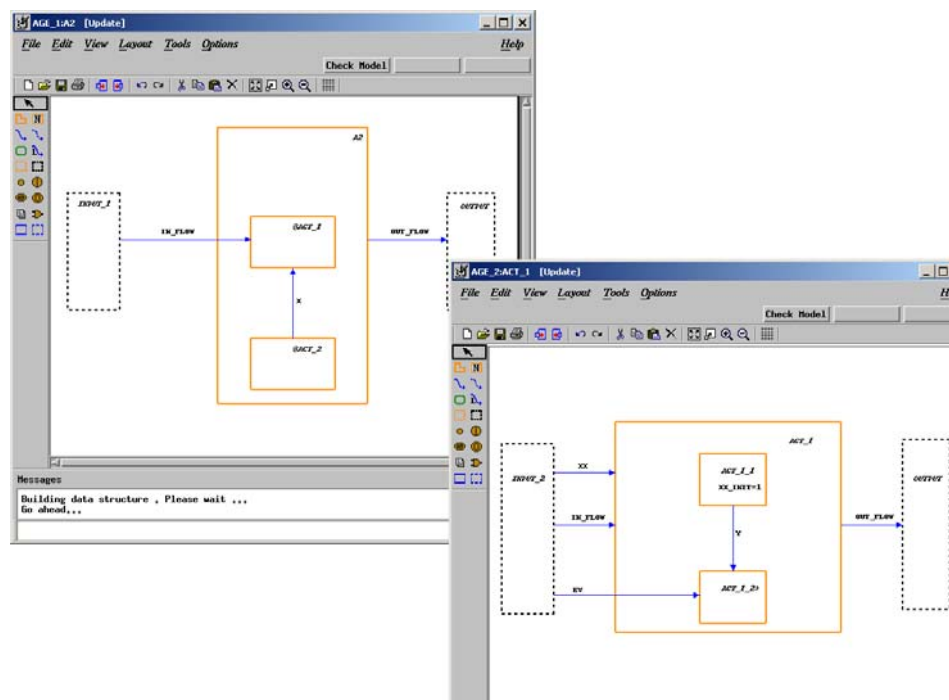


**Erroneous Elements:**

Events, conditions, data-items, and information-flows

# (C3059) Generic instances with inconsistent parameter binding

**Error In:**

Correctness

**Description:**

There is inconsistency between the actual bindings in the instance box and the defined, formal parameters in the generic chart.

**Example:**

A generic activity chart has a formal parameter x, which is defined as an integer. An activity chart contains an instance of this generic chart and has an actual parameter x1, which is bound to x. The actual parameter x1 is defined as a condition and is, therefore, inconsistent with the formal parameter definition.

**Erroneous Elements:**

Instance boxes

# (C3061) Controlled-termination activities that contain stop action

**Error In:**

Correctness

**Description:**

There are activities of reactive-controlled termination type, with STOP (sp!) action in their mini-spec.

**Erroneous Elements:**

Activities

# (C3063) Self-termination activities that do not contain stop action

**Error In:**

Correctness

**Description:**

There are activities of reactive-self termination type, but there is no STOP (sp!) action in their mini-spec.

**Erroneous Elements:**

Activities

# (C3065) Combinational elements aliases of registered elements or vice versa

**Error In:**

Correctness

**Description:**

An alias data-item is used as a combinational element and assigned a value in a combinational assignment, but the element that appears in its definition is used as a registered element, or vice versa.

**Example:**

In the following figure:

- ◆ XX is defined as a bit-array from 1 to 10.
- ◆ XX_INIT is defined as a single bit alias of XX defined as XX(1).
- ◆ Activity ACT_1_1 contains the following combinational assignment:

  XX_INIT=1

- ◆ Activity ACT_1_2 contains the following mini-spec:

  EV/XX(1)=1

The error is reported on XX_INIT.
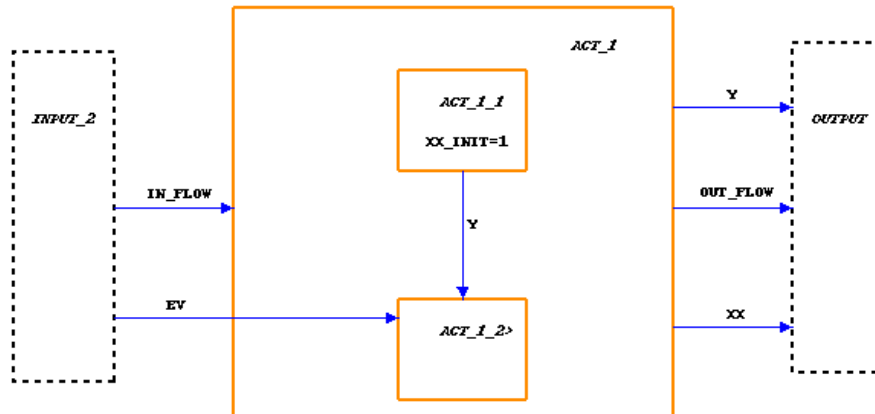
**Erroneous Elements:**

Data-items

# (C3067) Combinational elements assigned a value multiple times in the same entity

**Error In:**

Correctness

**Description:**

A combinational assignment is assigned a value in two separate, combinational assignments in the tested activity.

**Erroneous Elements:**

Data-items

# (C3069) Elements using field references that do not appear in the record / union definition

**Error In:**

Correctness

**Description:**

There are elements that contain expressions in which a field of an explicitly defined record or union is used, but the field does not appear in the definition of the record or union.

**Example:**

A record data-item `REC_1` flows from activity `ACT_1`. `REC_1` is defined to have two fields, `F1` and `F2`, both of which are integers. The activity `ACT_1` contains the following mini-spec:

```
EV/REC.F3 = 4
```

The error occurs because `F3` does not appear in the definition of `REC_1`.

**Erroneous Elements:**

Activities and textual elements

# (C3070) Elements erroneously using field references

**Error In:**

Completeness

**Description:**

Elements erroneously reference fields.

**Erroneous Elements:**

Any element referencing fields.

# (C3071) Formal parameters whose data-type is not global

**Error In:**

Correctness

**Description:**

Formal parameters are defined in terms of a User-Defined Type (UDT), but the UDT is defined in the generic chart. Because the UDT is not visible outside the generic chart, any actual binding cannot have this type. The error is generated in the tested generic chart.

**Note**

UDTs are normally defined in a Global Definition Set (GDS).

**Erroneous Elements:**

Data-items with formal parameters

# (C3073) Actual binding whose data-type is not global

**Error In:**

Correctness

**Description:**

Data-items, used as actual bindings in an instance box, are defined to be of some User-Defined Type (UDT), but the UDT is not global. It is defined in some chart, not in a GDS. Because this UDT is not visible to the generic chart, a formal parameter cannot have this type.

| Note |
| --- |

UDTs are normally defined in a GDS.

**Erroneous Elements:**

Data-items

# (C3075) Empty Records/Unions

**Error In:**

Correctness

**Description:**

There are data-items or data-types defined to be a record or union, but there are no fields defined in the structure. (This is a correctness error because it is a fatal error situation for some tools.)

**Erroneous Elements:**

Data-items and user-defined types

# (C3077) Non-generic instance blocks with explicit ports

**Error In:**

Correctness

**Description:**

**There are blocks that are not instances of generic charts and have explicit ports.**

**Erroneous Elements:**

**Blocks**

# (C3079) Blocks containing multiple ports with same name

**Error In:**

Correctness

**Description:**

Instance blocks of generic charts have more than one port with the same name.

**Erroneous Elements:**

Blocks

# (C3081) Bus/Signal flowing between two "in ports"

**Error In:**

Correctness

**Description:**

There is a bus or signal flowing between two "in" ports.

**Erroneous Elements:**

The components of the erroneous buses or signals

# (C3083) Bus/Signal flowing between two "out ports"

**Error In:**

Correctness

**Description:**

There is a bus or signal flowing between two "out" ports.

**Erroneous Elements:**

The components of the erroneous buses or signals

# (C3085) Sub-types that conflict with enumerated type definition

**Error In:**

Correctness

**Description:**

One of the following situations:

- ◆ Values that do not appear in the definition of an enumerated type are used.
- ◆ Values in the subtype definition are not in the same order as in the definition of the enumerated type.
- ◆ Values are missing from the subtype definition.

**Format:**

```
<subtype name> (sub range of <enumerated-type name>):
<enum-value> - missing in <enumerated-type name>
<enum-value> - missing in <subtype name>
<enum-value> - incorrect ordinal in <subtype name>
```

**Example:**

```
WORKING_DAYS (sub range of DAYS)
MONDAY - missing in DAYS
TUE - missing in WORKING_DAYS
WED - incorrect ordinal in WORKING_DAYS
```

**Erroneous Elements:**

User-defined types

# (C3087) Tasks/procedures that are used incorrectly

**Error In:**

Correctness

**Description:**

Tasks can be bound only to activities. Any other usage, including expressions, is incorrect. Procedures and functions cannot be bound to activities.

**Format:**

```
<subroutine name> (defined as <task/procedure/ function>) used as <procedure/
function/task> in definition of <element name>
```

**Example:**

A procedure PROC_1 is bound (using subroutine binding) to a reactive activity ACT_1.

**Erroneous Elements:**

Subroutines

# (C3089) Elements containing out-of-range assignments of enumerated values

**Error In:**

Correctness

**Description:**

An element of some subtype is assigned a value that belongs to the enumerated type, but is not in the range of the subtype.

**Format:**

```
in <definition/mini-spec...> of <element name>
<the assignment expression>
```

**Example:**

An enumerated type `DAYS` is defined as {`MON`,`TUE`,`WED`,`THU`,`FRI`,`SAT`,`SUN`}. `WEEKEND` is defined as a subtype of `DAYS` {`SAT`,`SUN`}. A mini-spec contains the following expression:

```
EV/WEEKEND=MON;
```

**Erroneous Elements:**

Textual elements, boxes, and arrows

# (C3091) Procedures/Functions that are used incorrectly

**Error In:**

Correctness

**Description:**

Procedures are used as functions or treated in some expression as if they return a value. Functions are used as procedures, or called in expressions where an action is expected.

**Format:**

```
<Function/Procedure> <subroutine name>
used as <Procedure/Function> in definition of
<element name> <expression of subroutine call>
```

**Example:**

Function FACTORIAL (returning an integer) is used as a procedure in the following mini-spec:

```
EV/FACTORIAL(X,Y)
```

**Erroneous Elements:**

Subroutines

# (C3093) External activities in generic chart with "User-Added Code Bindings"

**Error In:**

Correctness

**Description:**

External activities defined in generic charts (or descendants of generic charts) have a subroutine binding to a task, function, or procedure. The error is reported in the tested generic chart.

**Format:**

```
<Function/Procedure> <subroutine name>
```

**Erroneous Elements:**

Activities

# (C3095) Callback procedures that are used/defined incorrectly

**Error In:**

Correctness

**Description:**

Procedures are used as callbacks, whose first parameter is either not of type string or of mode Input.

**Format:**

```
<subroutine name> <parameter name>,
is an <"Output"/"InOut"> parameter
```

**Example:**

An activity `ACT_1` has a callback binding to a subroutine defined as a procedure:

```
C_BACK(X1,Y1)
```

The subroutine is defined with the following parameters:

- ◆ `X` - A string output parameter
- ◆ `Y` - An integer output parameter

The error is generated because the first parameter, `X1`, should be an input string.

**Erroneous Elements:**

Subroutines

# (C3097) Elements containing subroutine calls with conflicting length of parameters

**Error In:**

Correctness

**Description:**

There are subroutine calls where the actual parameter's length conflicts with the corresponding formal parameter definition, or whose length is unknown.

**Format:**

```
<element name>
<expression of subroutine call>
<actual parameters> - Unknown length
```

**Example:**

Activity `ACT_1` contains the following mini-spec:

```
EV/SUB_LEN(X1,Y1)
```

In this mini-spec:

- ◆ `X1` is an unbounded array of integers.
- ◆ `Y1` is a string.
- ◆ `SUB_LEN` is a subroutine with the following parameters:
    - `X` - An array of integers, from 1 to 5
    - `Y` - A string

The error is generated because `X1` is of unknown length.

**Erroneous Elements:**

Textual elements, boxes, and arrows

# (C3099) Subroutines using recursive calls

**Error In:**

Correctness

**Description:**

Subroutines use recursive calls, either directly or by other subroutines.

**Format:**

```
<subroutine name1> called from <subroutine name2>
<subroutine name2> called from <subroutine name1>
```

**Example:**

```
SR3 called from SR2
SR2 called from SR3
```

**Erroneous Elements:**

Subroutines

# (C3101) Subroutines that are used incorrectly

### Error In:

Correctness

### Description:

Subroutines use elements that are not declared as parameters, local variables, or globals. All elements used in the subroutine code should be declared, including constant data-items or conditions, enumerated values, other subroutines, and user-defined types.

### Format:

```
subroutine <subroutine name>
<used element name>
```

### Erroneous Elements:

Subroutines

# (C3105) Subroutines that contain references to activities

**Error In:**

Correctness

**Description:**

Subroutines contain references to activities. This can happen only by using a global variable that contains a reference to an activity.

**Format:**

```
subroutine <subroutine name> Activity <activity name>
used in <global variable name>
```

**Example:**

```
    subroutine F
    Activity A used in GLOBAL1
```

**Erroneous Elements:**

Subroutines

# (C3107) Subroutines with illegal parameter type or return type

**Error In:**

Correctness

**Description:**

A subroutine uses an illegal parameter type or return type. The following table lists the illegal definitions for UDTs.

| Type | Return Type | Parameter |
|---|---|---|
| **UDT array of string** | (*) | ** Illegal ** |
| **UDT array of bit-array** | (*) | ** Illegal ** |
| **UDT array of direct record or union** | (*) | ** Illegal ** |
| **UDT queue of anything** | ** Illegal ** | ** Illegal ** |
| **Array of queue (any dimension)** | ** Illegal ** | ** Illegal ** |

(*) In these cases, an interim UDT is needed. For example, if the return type of a function should be an array of string, it should be defined as `UDT array of STR_UDT`, and `STR_UDT`, a string.

**Format:**

```
subroutine <subroutine name>
return type <user-def type>, defined as <type> <user-def type>, type of
parameter <parameter>, defined as <type>
```

**Example:**

A subroutine, defined as a function, has a return type `T_ARR_STR`. `T_ARR_STR` is defined as an array of strings from 1 to 5. The error is generated because `T_ARR_STR` is defined directly. To correct this error, an intermediate UDT is required:

```
T_ARR_STR is defined as an array from 12 to 5 of UDT T_STR

T_STR is defined as a string
```

**Erroneous Elements:**

Subroutines

# (C3109) Subroutines with illegal local variables type

**Error In:**

Correctness

**Description:**

Local variables are defined as a user-defined type (UDT) that is a queue or an array (of any dimension) of a queue.

**Format:**

```
subroutine <subroutine name>
<user-def type>, type of local variable
<local variable>, defined as <type>
```

**Example:**

A subroutine F1, defined as a procedure, has a local variable LV1 defined in terms of a UDT T_LV1. T_LV1 is defined as a queue of integers, which is an illegal type for local variables.

**Erroneous Elements:**

Subroutines

# (C3111) Subroutines with illegal/missing RETURN command

**Error In:**

Correctness

**Description:**

There are functions without a RETURN command, procedures with a RETURN command, or a function with a return type that conflicts with the returned value's type.

**Erroneous Elements:**

Subroutines

# (C3113) Subroutines that call other subroutines whose global elements are missing from the calling subroutine definition

**Error In:**

Correctness

**Description:**

Global elements of a subroutine, SUB, must be declared as global elements of any other subroutine calling SUB.

**Erroneous Elements:**

Subroutines

# (C3115) Subroutines with local variables/parameters overriding parameters/globals

**Error In:**

Correctness

**Description:**

There are subroutines with local variables that override parameters or globals, or parameters overriding globals.

**Erroneous Elements:**

Subroutines

# (C3117) Subroutines with illegal combination of type and implementation

**Error In:**

Correctness

**Description:**

There are subroutines with an illegal combination of type and implementation.

**Example:**

A subroutine is defined as a task, having an implementation of procedural statechart. Procedural statecharts can be only the implementation of either a function or a procedure.

**Erroneous Elements:**

Subroutines

# (C3119) Elements containing a truth table with illegal factorization structure

**Error In:**

Correctness

**Description:**

Either joined input cells are not nested in a previous join or there are two or more rows that share the same input section, but have different output sections.

**Example:**

The following figure show the truth table definition for activity ACT_1. An error is generated because the joined cells in column 2 are not a subset of any previous join. The tool will warn of this situation when you save the truth table.

**Erroneous Elements:**

Subroutines, actions, and activities

# (C3121) Elements containing a truth table with illegal combination of outputs

**Error In:**

Correctness

**Description:**

Elements described by truth tables use the same element twice as an output, via aliases.

> **Note**
>
> This check ensures that the same element is not used twice as a whole. The check does *not* compare overlapping of slices.

**Example:**

The following figure shows the truth table definition for activity ACT_1. CO_2 is defined to be an integer. AL_CO_2 is defined to be an alias of CO_2. An error is generated because CO_2 is the same as AL_CO_2.



**Erroneous Elements:**

Subroutines, actions, and activities

# (C3123) Elements containing a truth table with illegal combination of inputs

**Error In:**

Correctness

**Description:**

Elements described by truth tables use the same element twice as an input, via aliases.

**Example:**

The following figure shows the truth table definition for activity `ACT_1`. `DATA_1` is defined to be an integer. `AL_DATA_1` is defined to be an alias of `DATA_1`. An error is generated because `DATA_1` is the same as `AL_DATA_1`.



**Erroneous Elements:**

Subroutines, actions, and activities

# (C3125) Elements containing out of range integer/real assignments

**Error In:**

Correctness

**Description:**

Elements contain assignments of integers, where an out-of-range value is assigned to a restricted range integer, or one restricted range integer is assigned to another where there is no intersection of ranges. It tests if an integer or real value is compared with a value outside its range, as well as values assigned to integer or real variable.

**Erroneous Elements:**

States, actions, data-items, and transitions

# (C3127) Activities that contain more than one control activity

**Error In:**

Correctness

**Description:**

Activities contain more than one control activity, via transparent activities.

**Example:**

As shown in the following figure, the top-level activity TOP contains a control activity CTL and subactivity A3. A3 is defined within its property as being transparent. The definition chart A3 also contains a control activity CTL_2. The error is generated because activity TOP effectively contains two control activities (because A3 is transparent).

**Erroneous Elements:**

Activities

# (C3129) Activities that contain sub-activities with the same name

**Error In:**

Correctness

**Description:**

Activities contain subactivities with the same name, via transparent activities.

**Example:**

As shown in the following figure, the top-level activity TOP contains subactivities A2 and A3. A3 is defined within its property as being transparent. The definition chart A3 also contains a subactivity A2. The error is generated because activity TOP effectively contains two subactivities with the same name (because A3 is transparent).
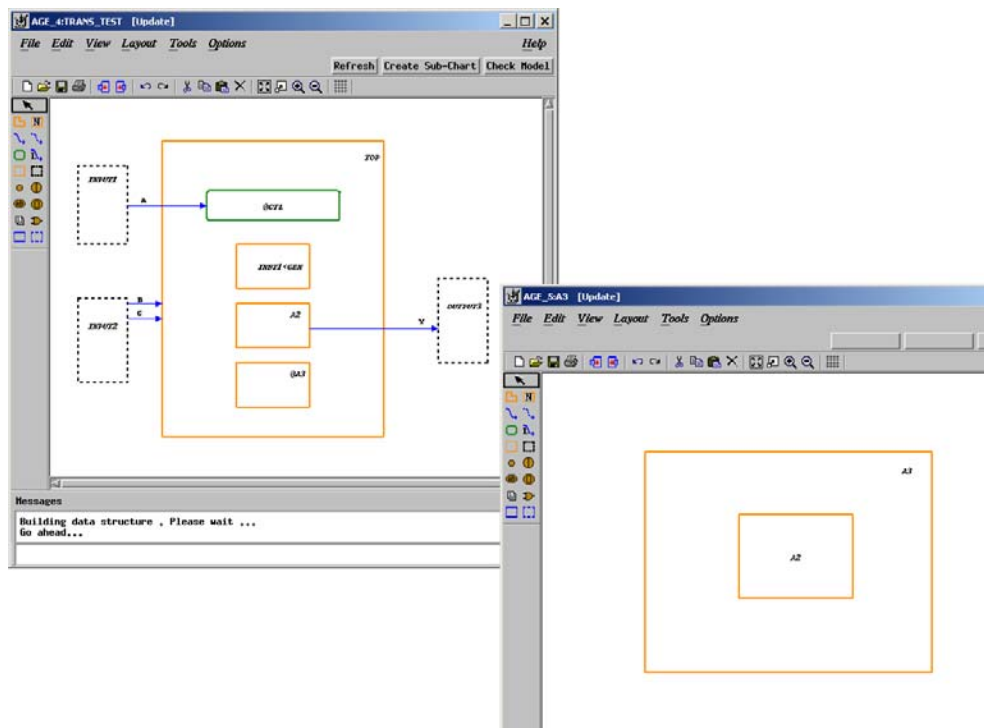


**Erroneous Elements:**

Activities

# (C3131) Illegal usage of components

**Error In:**

Correctness

**Description:**

Illegal usage of components.

**Erroneous Elements:**

Components

# (C3133) Instance components with illegal port bindings

**Error In:**

Correctness

**Description:**

A component instance activity has a port connected to an element of incorrect type.

**Example:**

A port of type integer data-item has been connected to a data-flow flowing to a string data-item.

**Erroneous Elements:**

Component instance activities

# (C3135) Continuous instances with inconsistent parameter binding

**Error In:**

◆ Correctness

**Description:**

There are continuous instances where the actual parameters are not consistent with the formal parameters of the definition continuous diagram. Parameter bindings might be missing, or parameter types might be inconsistent.

**Example:**

In the figure, A22<A22 is a continuous instance with actual parameters INT_1, INT_2, and INT_3, all integers. As shown in the properties for this instance, the actual parameter INT_3 is not bound to formal parameter I3, and an error is generated. Similarly, binding I1 (a real) to INT_1 (an integer) also generates this warning.
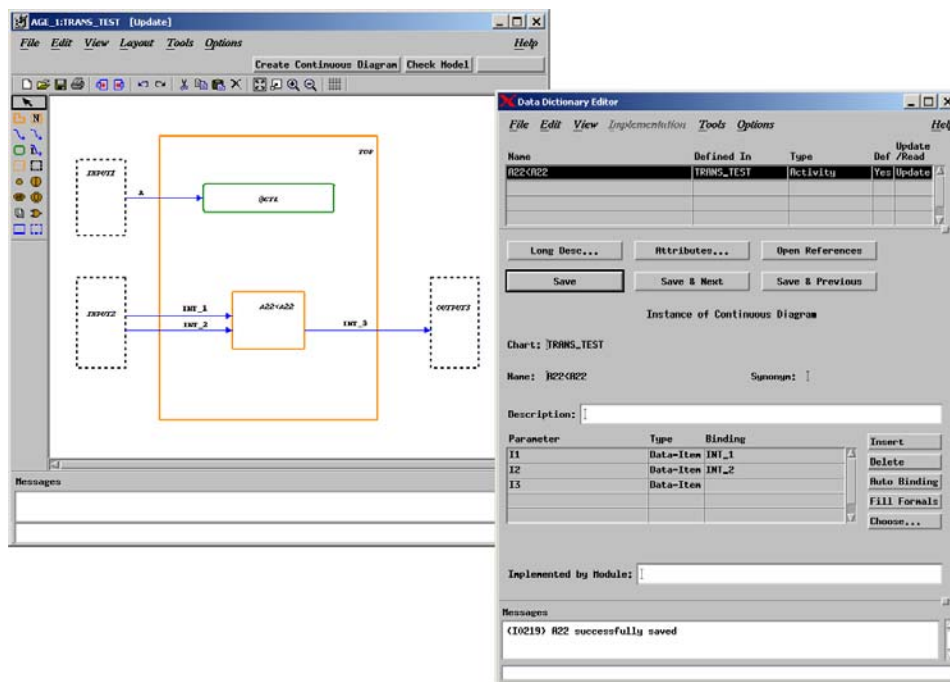


**Erroneous Elements:**

Continuous instances

# (C3137) Illegal switch structure contains conditional break

**Error In:**

Correctness

**Description:**

Switch/Case structures within an activity's mini-spec have a conditional `break` action.

**Example:**

Activity `ACT_1` contains the following mini-spec:

```
ch(SWITCH_VAL)/switch_c (SWITCH_VAL) {
 case_c 1: OUT1=RED;break;
 case_c 2: OUT1=AMBER;break;if COND1 then break;end if;
 case_c 3: OUT1=GREEN;break;
 default  : OUT=AMBER;}
```

The error is generated because there is a conditional break action statement for `case_c=2`.

**Erroneous Elements:**

Switch/Case structures

# (C3138) Switch/Case structure contain more than one default entry

**Error In:**

Correctness

**Description:**

Switch/Case expression with more than one "default" entry.

**Example:**

The following Rational Statemate Language expression is illegal:

```
/switch_c (DI) {
    case_c 1:
        DI++;
        break;
    default :
        DI--;
        break;
    default :
        DI--;
        break;
}
```

**Erroneous Elements:**

Activities, States, Action-Box, Transition, Action

# (C3138) Switch/Case structure contains more than one default entry

**Error In:**

Correctness

**Description:**

Switch/Case structures within an activity's mini-spec have more than one `default` statement.

**Example:**

Activity `ACT_1` contains the following mini-spec:

```
(SWITCH_VAL)/switch_c (SWITCH_VAL) {
    case_c 1: OUT1=RED;break;
    case_c 2: OUT1=AMBER;break;
    case_c 3: OUT1=GREEN;break;
    default : OUT1=RED;
    default  : OUT=AMBER;}
```

The error is generated because there is more than one `default` statement.

**Erroneous Elements:**

Switch/Case structures

# (C3139) Illegal switch structure: non-unique selected entries

**Error In:**

Correctness

**Description:**

Switch/Case structures within an activity's mini-spec have non-unique entries.

**Example:**

Activity `ACT_1` contains the following mini-spec:

```
ch(SWITCH_VAL)/switch_c (SWITCH_VAL) {
    case_c 1: OUT1=RED;break;
    case_c 2: OUT1=AMBER;break;
    case_c 2: OUT1=GREEN;break;
    default  : OUT=AMBER;}
```

The error is generated because two statements reference `case_c=2`.

**Erroneous Elements:**

Switch/Case structures

# (C3141) Inconsistent initial value

**Error In:**

Correctness

**Description:**

Checks that the default or initial value of integer and real data-items is consistent with the minimum and maximum value definitions. Note that this error is likely to occur when constructing data definitions outside of Rational Statemate (for example, when using Dataport). The tool checks that no default value outside the minimum and maximum ranges can be entered using the GUI.

**Example:**

An integer data-item is defined with a minimum value of 1, maximum value of 5, and an initial or default value of 7.

**Erroneous Elements:**

Integer and real data-items

# (C3143) Inconsistent range definition

**Error In:**

Correctness

**Description:**

This test warns about user-defined type definitions where the sub-type ranges are inconsistent with the parent type definition.

**Example:**

INT_TYPE is an integer User Defined Type and has a defined minimum/maximum range of 10 - 100 (as shown in the first figure below). SUB_INT_TYPE is a User Defined Type of INT_TYPE and has a range defined outside of 10 - 100 (as shown in the second figure below).
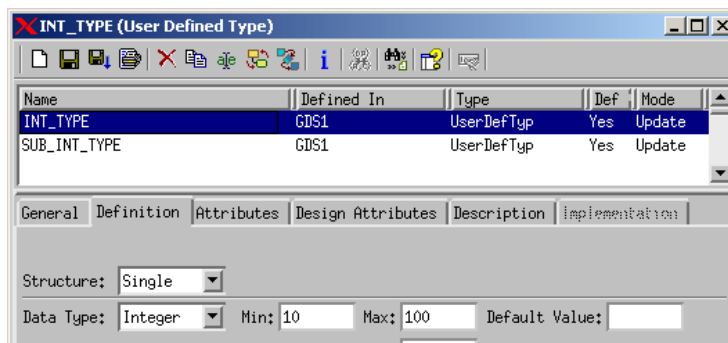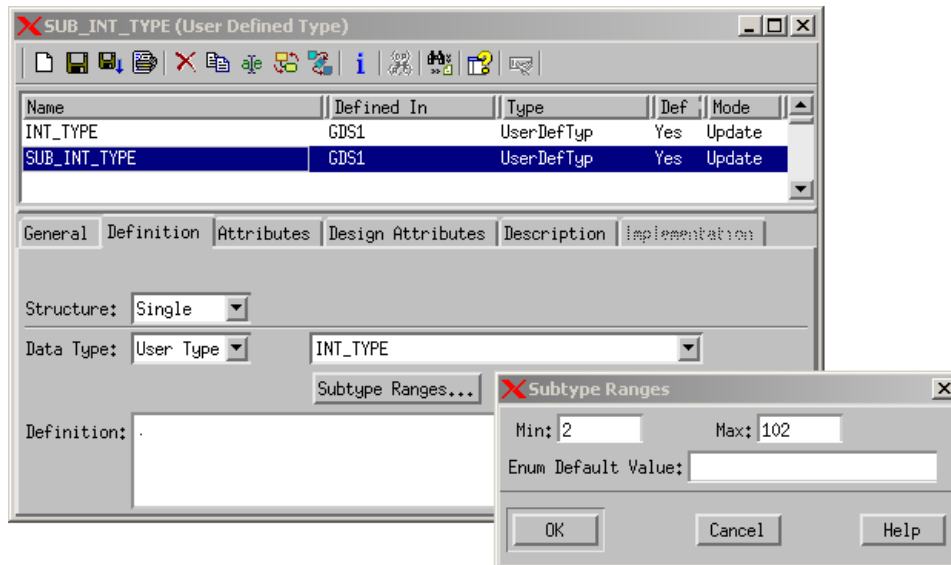
**Erroneous Elements:**

  User-defined types

# (C3145) Empty information-flows

### Error In:

Correctness

### Description:

Array index expression does not result in an integer or contain elements with illegal type.

### Example:

**Case I:**

Arr2 is an integer array from LeftInd2 to RightInd2.

LeftInd2 is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef2

SubDef2 is defined as:

Single None Constant with definition set to 2.34

**Case II:**

Arr1 is an integer array from LeftInd to RightInd.

LeftInd is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef

SubDef is defined as:

Single String Constant with definition set to 'My Definition'

# (C4001) Empty information-flows

**Error In:**

Completeness

**Description:**

There are information-flows whose Consists of field in the properties is empty.

**Erroneous Elements:**

Information-flows

# (C4003) Missing actual values for parameters

**Error In:**

Completeness

**Description:**

There is a generic instance activity in whose properties there are formal parameters, but no actual binding.

**Erroneous Elements:**

Instance activities

# (C4005) Elements with incomplete attribute definitions

**Error In:**

Completeness

**Description:**

There are elements in whose properties there is an attribute name without a corresponding attribute value, or vice versa.

**Erroneous Elements:**

Elements of all types

# (C4006) Elements with non-unique attribute definitions

### Error In:

Completeness

### Description:

Element contains more than one attribute with the same name.

### Erroneous Elements:

Any element with attributes

# (C4007) Elements that are not referenced

### Error In:

Completeness

### Description:

Textual elements are explicitly defined in the properties, but there is no reference to them in the model.

### Erroneous Elements:

Textual elements

# (C4009) Elements using redundant context variables

**Error In:**

Completeness

**Description:**

Elements contain actions with context variables that are assigned values, but not used in the action.

**Example:**

```
TICK/$I=0;$J=0;
for $J in 1 to 60 loop
    ARRAY($J)=0;
end loop;
```

**Erroneous Elements:**

Activities and actions

# (C4011) Elements that could be defined lower in the hierarchy of charts

**Error In:**

Completeness

**Description:**

Textual elements explicitly defined in the tested chart are used only in other charts, and could be defined lower in the chart hierarchy without changing the resolution results. The lowest chart in which these could be defined is the lowest common ancestor of all the charts in which references to this element exist.

**Example:**

Consider the following hierarchy of charts:

An element defined in SMS_SYSTEM but used only in PERFORM_RELEASE_GEN and its descendants should be defined in PERFORM_RELEASE_GEN.

**Erroneous Elements:**

Textual elements
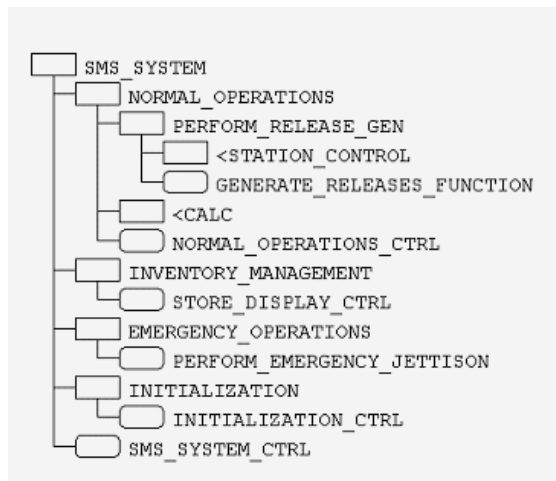
# (C4013) Elements containing references to components of non-array elements

**Error In:**

Completeness

**Description:**

Elements contain data-item expressions with indices, but the accessed data-item is not an array.

**Erroneous Elements:**

Actions, activities, conditions, and events

# (C4015) Elements containing comparisons of unions: (not supported in Code Generation

**Error In:**

Completeness

**Description:**

Elements contain comparisons of two union data-items. The comparisons of two unions is ambiguous, because the result depends on which field of the unions is the current field in each.

**Erroneous Elements:**

Actions, activities, conditions, and events

# (C4017) Unlabeled compound flow-lines

**Error In:**

Completeness

**Description:**

There are compound flow-lines that have unlabeled components.

**Example:**

Consider the following activity chart.
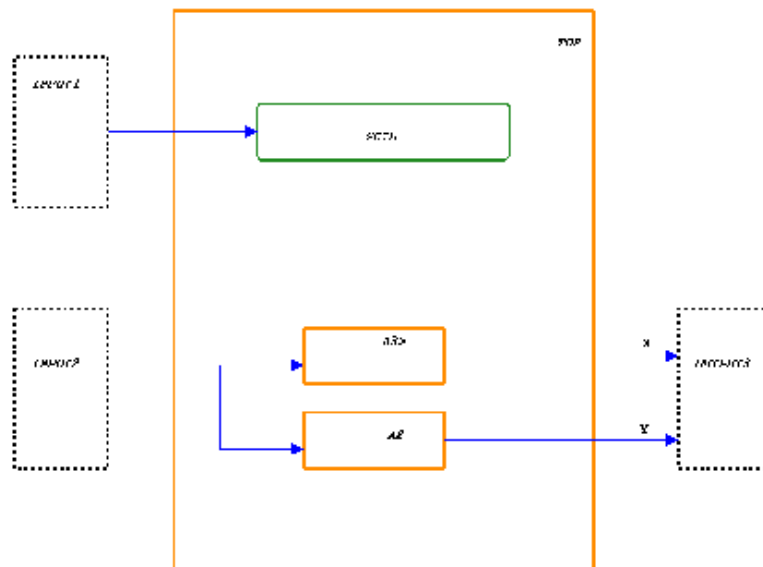


**Erroneous Elements:**

Components of erroneous flow-lines

# (C4019) Unlabeled flow-lines entering or exiting composition connectors

### Error In:

Completeness

### Description:

There are flow-line segments whose source or target is a composition connector and to which no label is attached.

### Erroneous Elements:

Unlabeled flow-line segments

# (C4020) Arrows not a component of any compound data-flow

### Error In:

Completeness

### Description:

This test is pointing out flow-lines in activity charts which are not recognized as valid flow-lines, and in some cases might cause other Check Model errors like C4058 and C4059.

### Erroneous Elements:

Flow Lines

# (C4021) Flow-lines with incomplete source definition

**Error In:**

Completeness

**Description:**

There are flow-lines whose source is not connected to an activity, data-store, or connector. The source of such flow-lines appears in the activity chart as a small line perpendicular to the flow-line.

**Erroneous Elements:**

Flow-lines

**Example:**
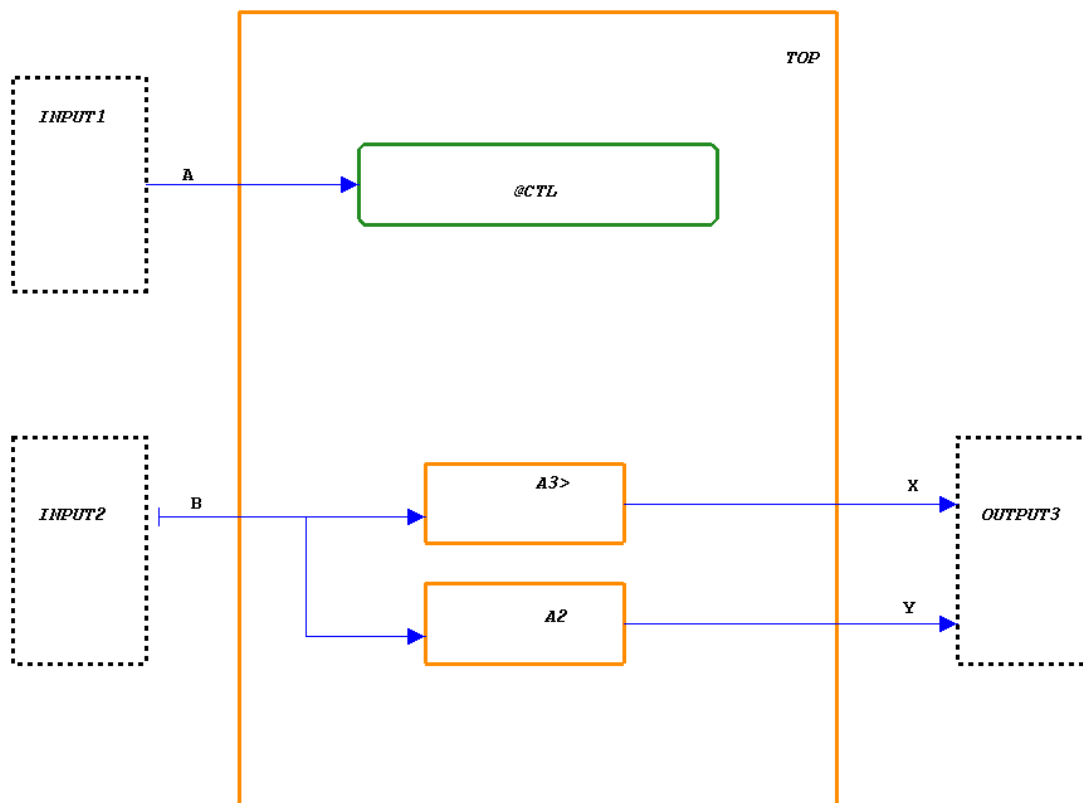
# (C4023) Flow-lines with incomplete target definition

**Error In:**

Completeness

**Description:**

There are flow-lines whose target is not connected to an activity, data-store, or connector. The target of such flow-lines appears in the activity chart as a small line perpendicular to the flow-line.

**Example:**

See the following figure.
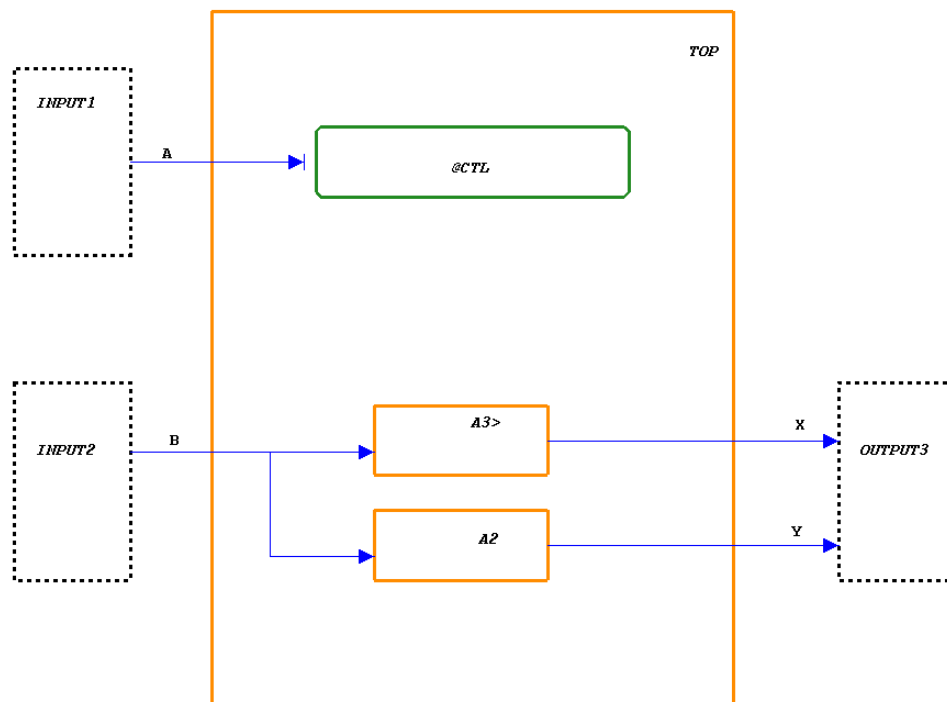


**Erroneous Elements:**

Flow-lines

# (C4025) Compound flow-lines with identical sources

**Error In:**

Completeness

**Description:**

There are compound flow-lines whose source set contains the same activity more than once, or an activity and one of its descendants.

**Example:**

Components of flow-line x in the following chart:
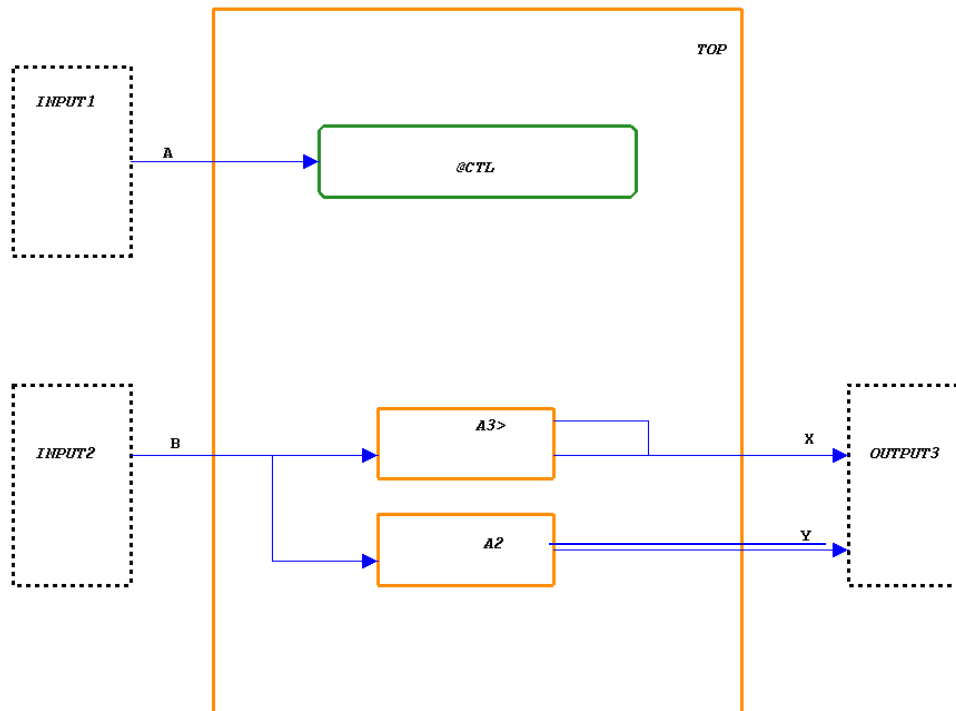
# (C4027) Compound flow-lines with identical targets

**Error In:**

Completeness

**Description:**

There are compound flow-lines whose target set contains the same activity more than once, or an activity and one of its descendants.

**Example:**

Components of flow-line B in the following chart.
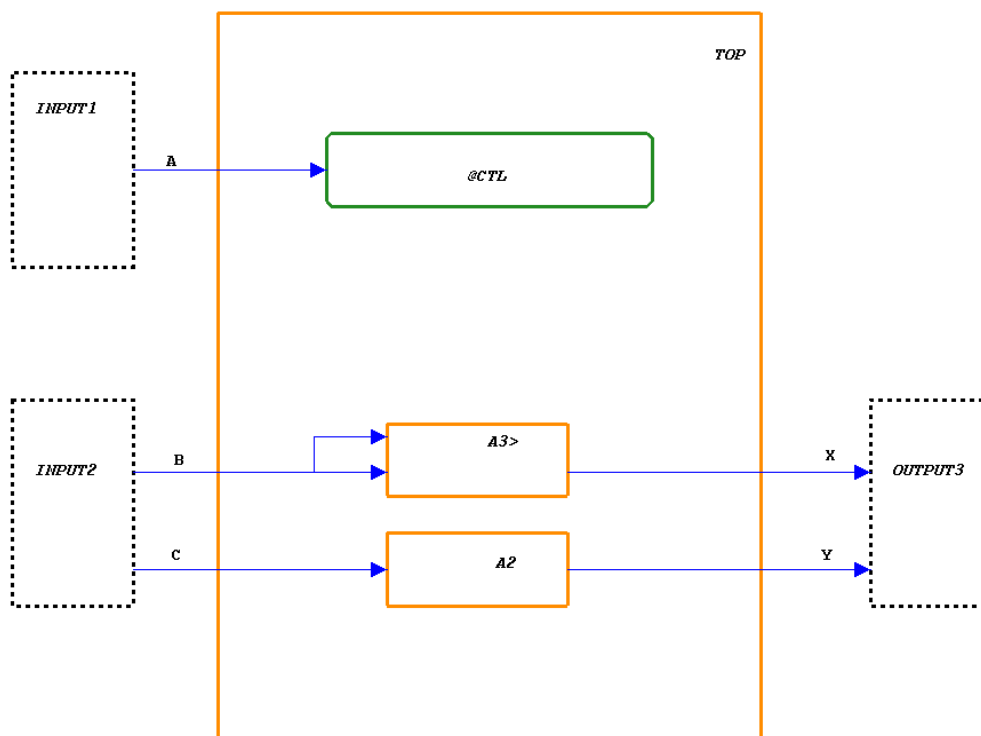


**Erroneous Elements:**

Components of compound flow-lines

# (C4029) Basic activities without any output

**Error In:**

Completeness

**Description:**

Basic activities and their ancestors have no outgoing flow-lines.

**Example:**

The output of activity A3 in the following chart. See the following figure.
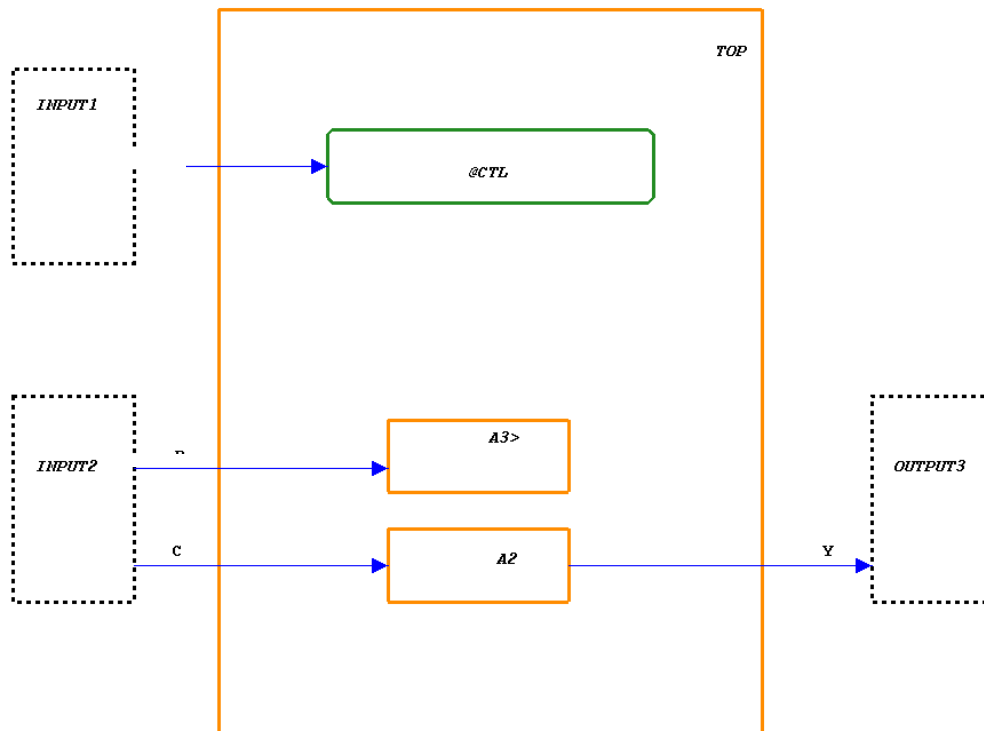


**Erroneous Elements:**

Activities

# (C4031) Data-stores without incoming flow-lines or without outgoing flow-lines

### Error In:

Completeness

### Description:

There are data-stores for which there are no input flow-lines, or there are no output flow-lines. Both inputs and outputs must exist.

### Example:

The data-store STORE has no input in the following chart. See the following figure.
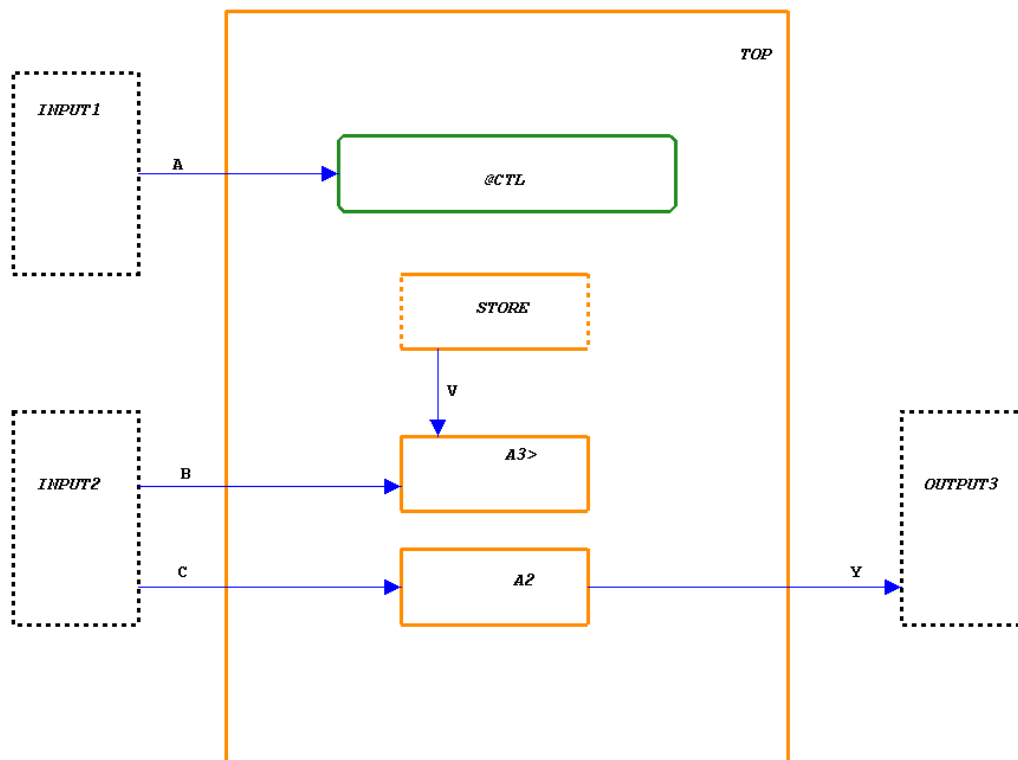


### Erroneous Elements:

Data-stores

# (C4033) Activities with a single subactivity or data-store

**Error In:**

Completeness

**Description:**

Activities contain a single subelement: an internal activity or a data-store. Activities that contain a control activity but no other subactivities are allowed.

**Erroneous Elements:**

Activities

# (C4035) Activities containing "to-control" connector without control-activity

**Error In:**

Completeness

**Description:**

There are activities in which a "to-control" connector is drawn, but do not contain a control activity.

**Erroneous Elements:**

Activities

# (C4037) Composition connectors with illegally located sources or targets

**Error In:**

Completeness

**Description:**

There are composition connectors whose components flow from sources or to targets that are outside the activity in which the connectors reside.

**Example:**
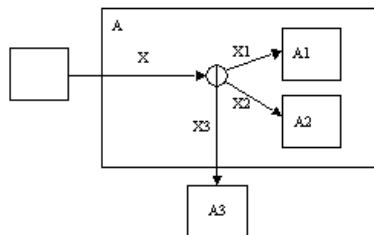


**Erroneous Elements:**

Connectors

# (C4039) Isolated connectors

**Error In:**

Completeness

**Description:**

There are connectors with no entrances and no exits.

**Erroneous Elements:**

Connectors

# (C4041) Connectors with an entrance and no exit

**Error In:**

Completeness

**Description:**

There are junction or composition connectors, or a group of in-page diagram connectors with the same name, with an entrance but no exit.

**Erroneous Elements:**

Connectors

# (C4043) Connectors with an exit and no entrance

**Error In:**

Completeness

**Description:**

There are junction or composition connectors, or a group of in-page diagram connectors of the same name, with an exit but no entrance.

**Erroneous Elements:**

Connectors

# (C4045) Junction or composition connectors with a single entrance and a single exit

**Error In:**

Completeness

**Description:**

There are junction or composition connectors with a single entrance and a single exit. Therefore, they are redundant.

**Erroneous Elements:**

Connectors

# (C4047) Composition connectors with missing flow elements

### Error In:

Completeness

### Description:

There are composition connectors whose record decomposition is not complete.

### Example:

In the figure, data-item B is defined in terms of a record having three fields - F1, F2, and F3. Field F3 is missing from the chart.
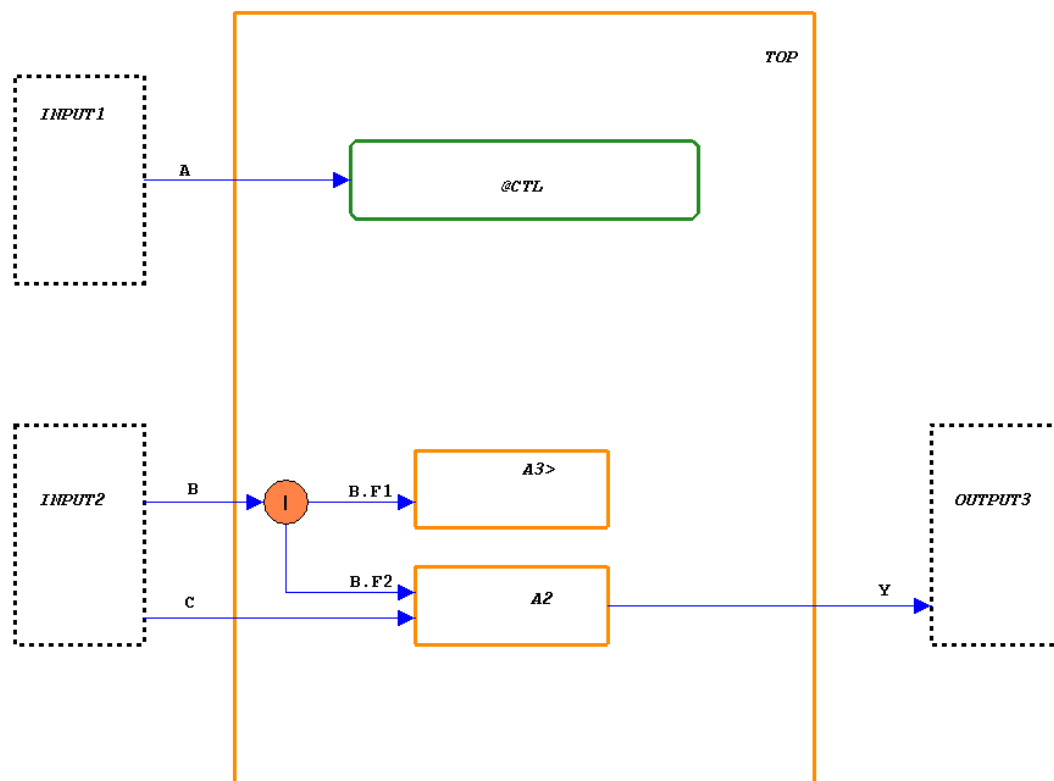


### Erroneous Elements:

Data-items

# (C4048) Unresolved elements

**Error In:**

Completeness

**Description:**

There are reference elements for which no element with the same name is explicitly defined in the relevant scope. This check also checks resolution of external activities in an off-page chart to activities, data-stores, or modules in the parent chart.

**Example:**

The following figure shows that the external activity E2 in the definition chart B has no matching external activity in the instance chart.
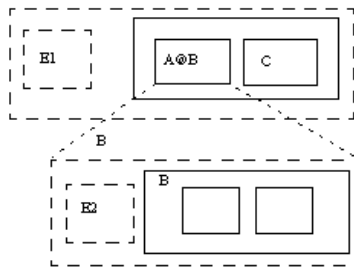


**Example 2:**

A data-item, used within the mini-spec of an activity, is not defined in the relevant scope. The data-item appears as textual in the properties.

**Erroneous Elements:**

All types of elements

# (C4049) The tested chart is a definition chart with multiple top-level boxes

### Error In:

Completeness

### Description:

The tested chart is used as an off-page definition of an instance module and has multiple top-level modules.

This check is performed only when you specify with definition charts. The error is reported in the definition chart.

### Erroneous Elements:

The tested chart

# (C4050) The tested chart is a definition chart with empty top-level box

### Error In:

Completeness

### Description:

Either the tested chart is used as an off-page definition of an instance activity and has a single, empty, top-level activity; or, a single activity chart is tested that has a single, empty, top-level activity.

This check is performed only when you specify with definition charts. The error is reported in the definition chart.

### Erroneous Elements:

The tested chart

# (C4051) Instance off-page connectors without a matching definition off-page connector

**Error In:**

Completeness

**Description:**

There are instance off-page connectors that have no matching definition diagram connector.

This check is performed only when you specify with definition charts. Both instance and definition charts should be included in the Check Model profile. The error is reported in the instance chart.

**Example:**

Considering the decomposition of activity A2, the diagram connector shown on the instance chart is not shown on the definition chart.

**Erroneous Elements:**

Tested chart
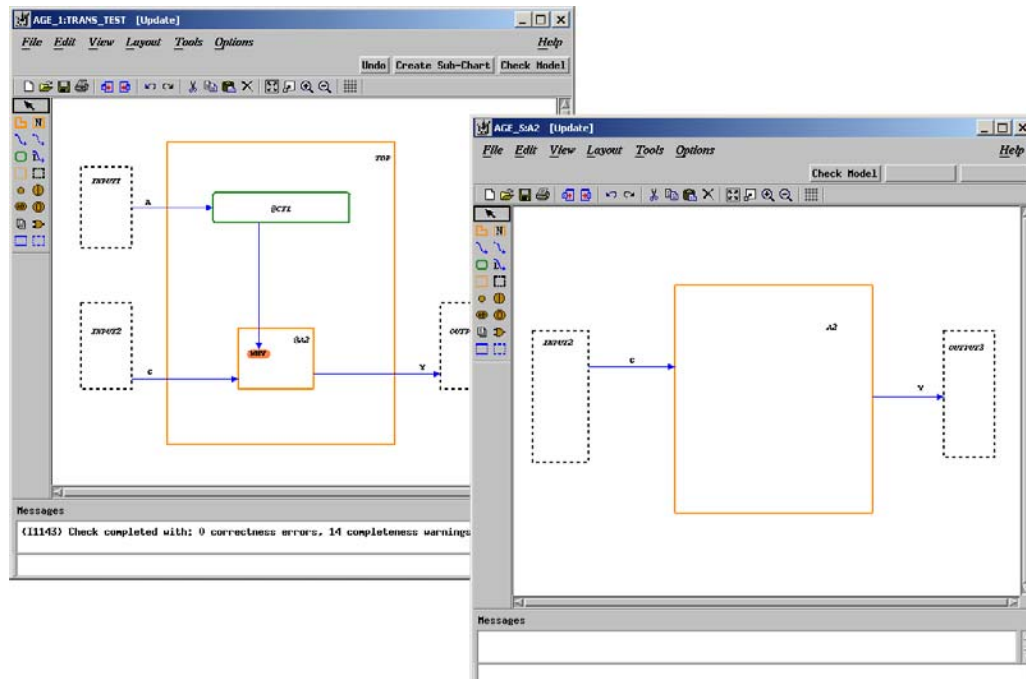
# (C4052) In parameters not used in the chart

**Error In:**

Completeness

**Description:**

There are Input or InOut parameters of a generic chart that are not used (as part of the trigger in any mini-spec or statechart) within the chart. The error is reported when performing a standalone test of the generic chart and when the chart containing the generic instance is tested. The error is reported in the generic definition chart.

**Erroneous Elements:**

Generic parameters - events, conditions, and data-items

# (C4053) Out parameters not affected (generated or modified) in the chart

**Error In:**

Completeness

**Description:**

There are Output or InOut parameters of a generic chart that are not affected (as part of the action in any mini-spec or statechart) within the chart. The error is reported when performing a standalone test of the generic chart and when the chart containing the generic instance is tested. The error is reported in the generic definition chart.

**Erroneous Elements:**

Generic parameters - events, conditions, and data-items

# (C4054) In parameters not used in the chart

**Error In:**

Completeness

**Description:**

There are Input or InOut parameters that are not used in the chart.

**Erroneous Elements:**

Textual elements and activities parameters

# (C4055) Out parameters not affected in the chart

**Error In:**

Completeness

**Description:**

There are Output or InOut parameters that are not affected (modified or generated) in the chart.

**Erroneous Elements:**

Textual elements and activities parameters

# (C4056) In parameters affected in the chart

**Error In:**

Completeness

**Description:**

There are Input or InOut parameters that are affected (modified or generated) in the chart.

**Erroneous Elements:**

Textual elements and activities parameters

# (C4057) Out parameters used in the chart

### Error In:

Completeness

### Description:

There are Output or InOut parameters that are used in the chart.

### Erroneous Elements:

Textual elements and activities parameters

# (C4058) Elements that flow from/to instance without a matching flow in definition chart

### Error In:

Completeness

### Description:

There are elements that flow from or to the instance activity without matching flow-lines to or from an external activity in the off-page definition chart. Note that the elements flowing on flow-lines are matched, *not* the sources or targets of the flow.

This check is performed only when you specify with definition charts. The error is reported in the instance chart.

### Example:

In the following figure, C is an input to the instance @A2. In the off-page definition chart A2, there is no matching flow from an external activity.
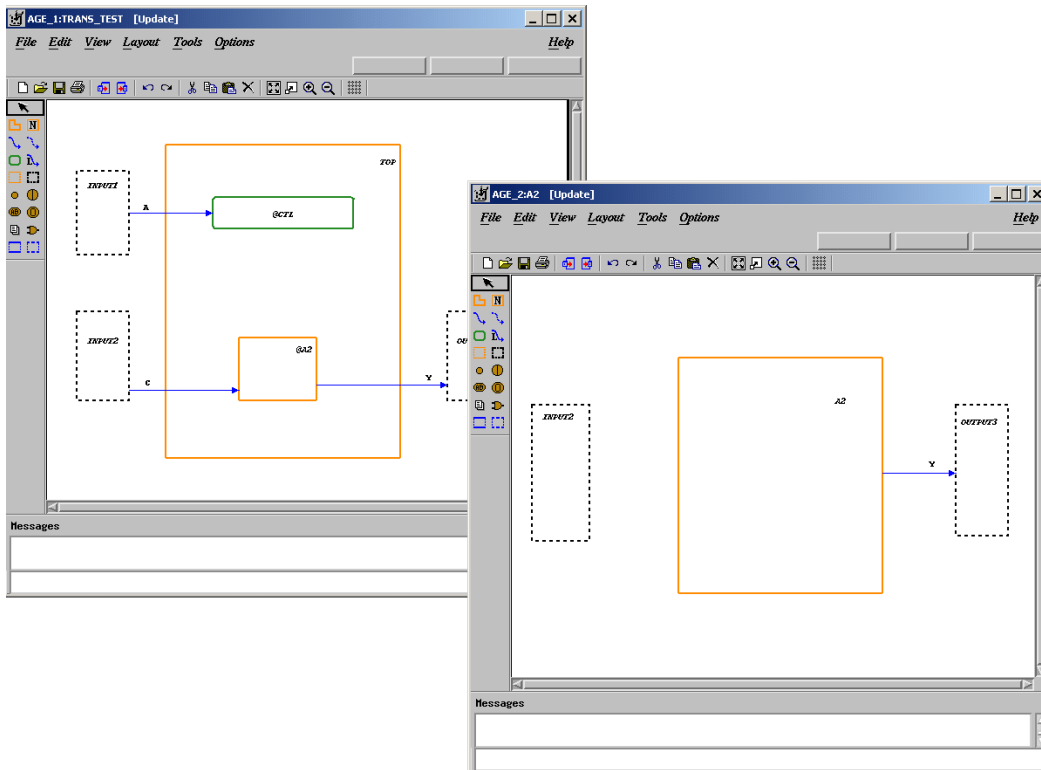
**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C4059) Elements that flow from/to definition without a matching flow in instance chart

**Error In:**

Completeness

**Description:**

There are elements that flow to or from an external activity in the definition chart without matching flow-lines from or to the instance. Note that the elements flowing on the flow-lines are matched, *not* the sources or targets of the flow.

This check is performed only when you specify with definition charts. The error is reported in the definition chart.

**Example:**

In the following figure, c is an input to an activity A2 in the off-page definition chart. However, in the chart containing the instance @A2, there is no matching flow.
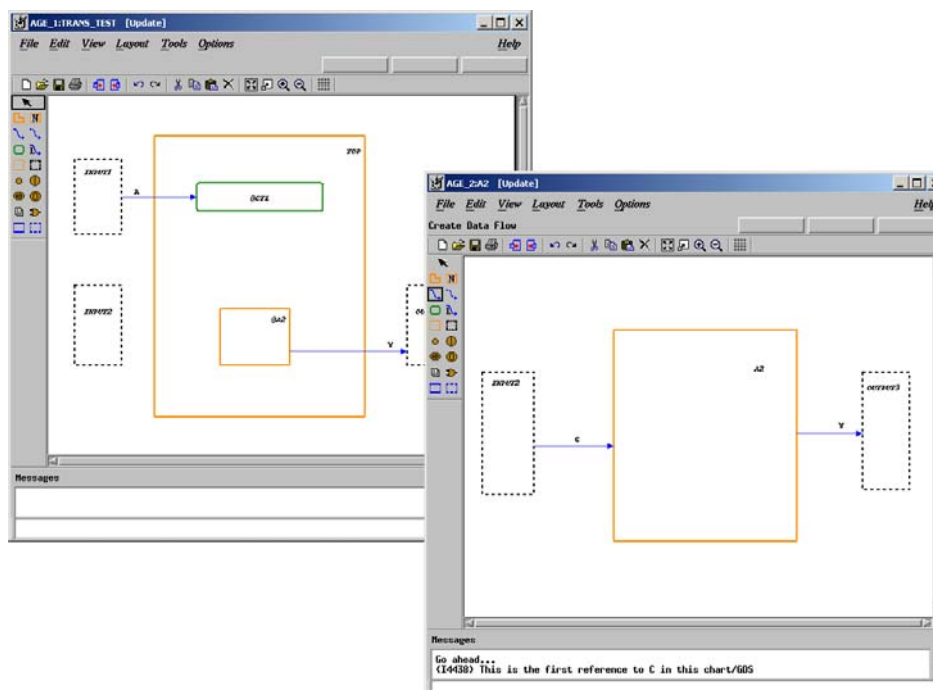


**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C4061) Elements that are input to an activity and output of another activity without a flow-line between the two activities

### Error In:

Completeness

### Description:

There are elements that flow on separate flow-lines.

If an activity that is a source of one flow-line changes the value of the element and this value is used by a target activity of the other flow-line, the drawing might be misleading.

### Example:

In Figure a, x is changed by activity A and is used by D, but there is no flow-line between A and D. Therefore, an error is generated.

However, the separate flow-lines in Figure b are allowed because they have the same source. Therefore, no error is generated.



(a) Error          (b) Correct
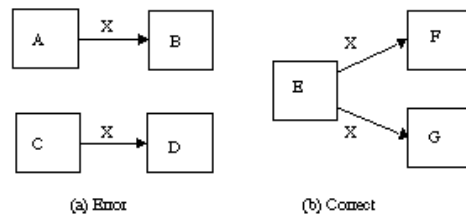
### Erroneous Elements:

Events, conditions, data-items, information-flows, and fields

# (C4063) Actual parameters not flowing to/from generic instance

**Error In:**

Completeness

**Description:**

There are actual parameters bound to some Input/Output formal parameter in an instance box, but are not shown graphically as being Input/Output to that box. Parameters must be defined.

**Example:**

In the following figure, the actual parameters `P1` and `S1` are bound to the formal parameters in the properties, but they are not shown as being input or output to the instance box in chart `A2`.



**Erroneous Elements:**

Events, conditions, data-items, and fields

# (C4065) Elements that flow to/from generic instance but are not used as actual parameters

**Error In:**

Completeness

**Description:**

There are elements that flow into or from an instance box of a generic chart, but are not declared as actual Input or Output parameters. Parameters *must* be defined.

**Example:**

In the following figure, the actual parameters P1 and S1 are shown as being input/output to the instance box in chart A2, but they have not been bound to the formal parameters in the properties.



**Erroneous Elements:**

Events, conditions, data-items, and fields

# (C4067) External elements not declared as formal parameters

**Error In:**

Completeness

**Description:**

There are elements that flow to or from an external activity, to or from the top-level activity in a generic chart, but do not appear as In or Out formal parameters of the generic chart.

**Erroneous Elements:**

Events, conditions, and data-items

# (C4069) Formal parameters not appearing on external flow-lines

**Error In:**

Completeness

**Description:**

There are elements declared as In or Out parameters of a generic activity chart, but they do not flow into or out of the top-level activity to an external box in the generic chart.

**Erroneous Elements:**

Events, conditions, and data-items

# (C4071) Elements affected by an activity but are neither used internally nor output

**Error In:**

Completeness

**Description:**

There are elements that are affected (modified or generated) in the mini-spec of an activity, but are neither used in this mini-spec nor as output of the activity.

**Example:**

In the figure, activity `INST2` has the following mini-spec:

    C/X=4;Y=2

`X` is not shown as an output on the chart and is not used internally within the mini-spec; however, it is affected by `INST2`.
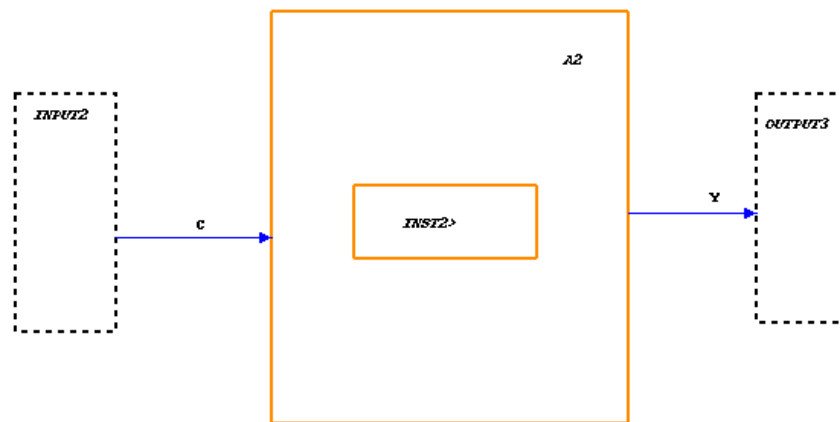


**Erroneous Elements:**

Events, conditions, and data-items

# (C4073) Elements that are used by an activity but are neither input nor affected internally

**Error In:**

Completeness

**Description:**

There are elements used in the mini-spec of an activity, but they are neither input nor affected (modified or generated) in the mini-spec.

**Example:**

In the figure, activity `INST2` has the following mini-spec:

```
C and D/X=4;Y=2
```

`D` is not shown as an input to the chart and is not affected internally within the mini-spec; however, it is used by `INST2`.
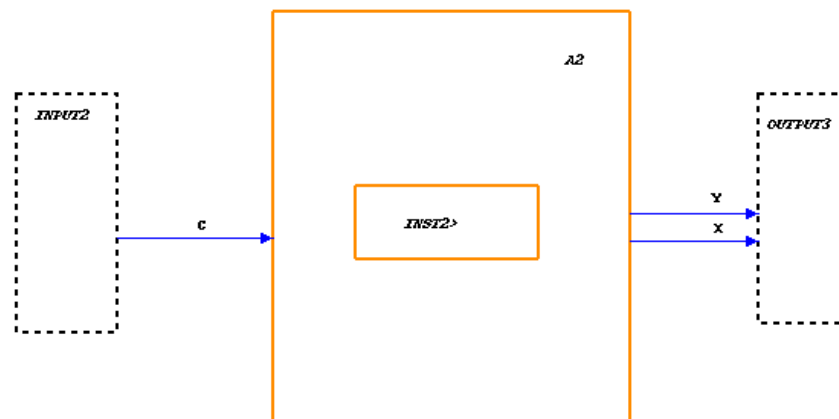


**Erroneous Elements:**

Events, conditions, and data-items

# (C4075) Elements output from activity that are not affected internally

**Error In:**

Completeness

**Description:**

There are elements that are output of an activity, but they are not affected (modified or generated) in its mini-spec.

**Example:**

In the figure, activity `INST2` has the following mini-spec:

```
C and D/Y=2
```

`X` is shown as an output from activity `INST2`, but is not affected within the mini-spec of `INST2`.
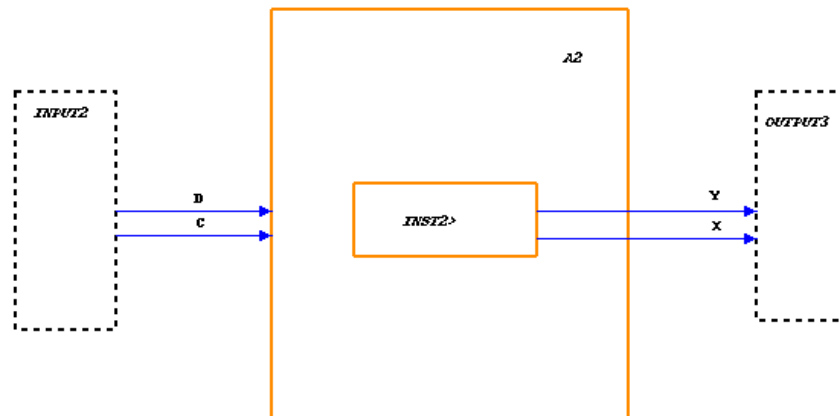


**Erroneous Elements:**

Events, conditions, and data-items

# (C4076) Flow of empty Information flow

### Error In:

Completeness

### Description:

There are Information-Flows with no Elements in their "Consist Of" matrix.

### Erroneous Elements:

low-lines

# (C4077) Elements input to activity that are not used internally

### Error In:

Completeness

### Description:

There are elements that are input to an activity, but they are not used in its mini-spec.

### Example:

In the figure, activity `INST2` has the following mini-spec:

```
C/X=4;Y=2
```

`D` is shown as an input to activity `INST2`, but is not used within the mini-spec of `INST2`.
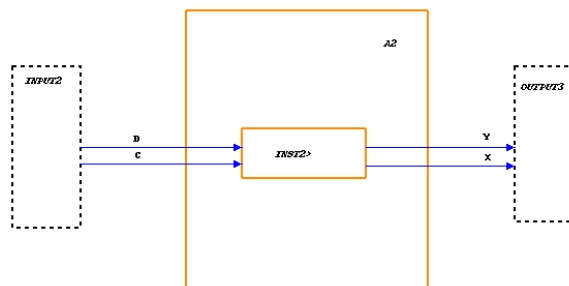


### Erroneous Elements:

Events, conditions, and data-items

# (C4079) Data elements with incomplete type definition

### Error In:

Completeness

### Description:

There are data elements whose type definition is missing or incomplete.

### Example:

Element x is defined within the chart A2. x is defined in terms of a user-defined type (UDT) T_X, where T_X has not been defined.

### Erroneous Elements:

Events, conditions, data-items, and UDTs

# (C4081) Constant parameter arrays with incomplete length definition

### Error In:

Completeness

### Description:

There are constant parameter arrays where either the definition of the formal parameter in the generic chart is incomplete, or they are bound to an actual parameter of undefined length.

### Erroneous Elements:

Data-items and conditions

# (C4083) Defined elements that can be merged with elements from ancestor charts

**Error In:**

Completeness

**Description:**

There are textual elements explicitly defined in a chart that have matching, explicitly defined elements with the same name in ancestor charts. Both elements have the same type in both charts; therefore, they could probably be merged.

**Erroneous Elements:**

Textual elements

# (C4085) Activities whose mini-specs contain `started' trigger or `stop' action (in HW style)

**Error In:**

Completeness

**Description:**

There are activities whose mini-specs contain started triggers or stop actions. This is improper in HW style, where all activities are always active.

**Erroneous Elements:**

Activities

# (C4087) Procedure-like or self-terminated activities (in HW style)

**Error In:**

Completeness

**Description:**

There are activities defined in the properties as procedure-like or reactive self-terminated activities. This is improper in HW style, where all activities are always active.

**Erroneous Elements:**

Activities

# (C4089) Activities with combinational assignments

**Error In:**

Completeness

**Description:**

There are activities with combinational assignments.

**Erroneous Elements:**

Activities

# (C4091) The tested activity chart contains no activities

**Error In:**

Completeness

**Description:**

The tested activity chart contains no activities.

**Erroneous Elements:**

The tested activity chart

# (C4093) Elements overriding global definitions

**Error In:**

Completeness

**Description:**

There are textual elements with the same name as elements defined in a global definition set.

**Erroneous Elements:**

Textual elements

# (C4095) Subroutines not implemented in any language

**Error In:**

Completeness

**Description:**

There are subroutines that have no implementation in the Rational Statemate Action Language, Procedural Statechart/Flowchart, Ada Code, K&R C Code, ANSI C Code, and so on.

**Erroneous Elements:**

Subroutines

# (C4097) Elements containing triggers that use functions with potential side effects

### Error In:

Completeness

### Description:

Functions are used in the trigger part of labels, static reactions, or mini-specs and have side effects (that is, they affect global data).

### Format:

```
<element name>
Function <function name> accessing global data, used in trigger: <trigger
expression>
```

### Example:

Activity `A1` has the following mini-spec, where `F1` is a function returning an integer that accesses global data (through Globals Usage in the properties):

```
EV and [F1(X)==6]/Y=4
```

### Erroneous Elements:

States, activities, and transitions

# (C4099) Textual subroutines using uninitialized local variables

**Error In:**

Completeness

**Description:**

Local variables of a textual subroutines are used before they are affected or assigned a value.

**Format:**

```
<local variable name> in <subroutine name>
```

**Example:**

Subroutine F1 is defined to be a function with input parameter X that returns an integer Y and has two local variables (A and B) defined:

```
A=B;
B=X+3;
Y=B;
return(Y)
```

In this case, B is used before it is initialized.

**Erroneous Elements:**

Subroutines

# (C4101) Textual subroutines using redundant local variables

**Error In:**

Completeness

**Description:**

Local variables of a textual subroutines are affected (modified or generated) but not used.

**Format:**

```
<local variable name> in <subroutine name>
```

**Example:**

Subroutine F1 is defined to be a function with input parameter X that returns an integer Y and has two local variables (A and B) defined:

```
A=B;
B=X+3;
Y=B;
return(Y)
```

In this case, A is a redundant local variable.

**Erroneous Elements:**

Subroutines

# (C4103) Subroutines containing parameters defined as a non-unique user-defined type

### Error In:

Completeness

### Description:

A parameter of a subroutine is defined as a user-defined type (UDT), whose name is not unique in the model.

### Example:

DAYS, a parameter of function F1 is defined in terms of a UDT T_DAYS. This UDT is defined twice in the model and therefore is not unique.

### Erroneous Elements:

Subroutines

# (C4105) Enumerated types with non-unique value names

### Error In:

Completeness

### Description:

There are enumerated types with values that appear in other unrelated enumerated type definitions (not in a subtype).

### Example:

The enumerated value BLUE exists in both UDTs COLORS and MOOD.

### Erroneous Elements:

User-defined types

# (C4106) Data-Elements with Enumerated Value names

### Error In:

Correctness

### Description:

Textual element has same name as enumerated values.

### Erroneous Elements:

Textual elements

# (C4107) Subroutines containing wrong usage of parameters or globals

### Error In:

Completeness

### Description:

Subroutines contain the wrong usage of parameters or globals. For example:

- In parameters that are affected by the subroutine.
- In parameters that are not used by the subroutine.
- Out parameters that are used by the subroutine.
- Out parameters that are not affected by the subroutine.
- InOut parameters that are not used by the subroutine.
- InOut parameters that are not affected by the subroutine

### Example:

`DAYS`, an input parameter of function `FUNC1`, is not used within the subroutine.

### Erroneous Elements:

Subroutines

# (C4109) Subroutines with incomplete type definition

**Error In:**

Completeness

**Description:**

Local variables are not used in the subroutine code.

**Example:**

DAYS, an input parameter of function FUNC1, is not defined within the model.

**Erroneous Elements:**

Subroutines

# (C4111) Local variable not used in the subroutine

**Error In:**

Completeness

**Description:**

Local variables are not used in the subroutine code.

**Format:**

```
subroutine <subrtn name> Local variable <local var name> not used
```

**Example:**

DAYS, a local variable of function FUNC1, is not used within the model.

**Erroneous Elements:**

Subroutines

# (C4113) Elements with illegal "Selected Implementation"

### Error In:

Completeness

### Description:

The selected implementation is not one of the existing implementations of the element.

### Example:

Subroutine function `FUNC1` is described with an implementation of action language. However, the Selected Implementation in the properties is set to procedural statechart, an implementation that does not actually exist.

### Erroneous Elements:

Subroutines

# (C4115) Elements containing a truth table with empty default row

### Error In:

Completeness

### Description:

Elements described by truth tables contain a default row with the output columns blank and the action section empty.

### Erroneous Elements:

Subroutines and actions

# (C4117) Elements containing a truth table with empty cells

**Error In:**

Completeness

**Description:**

Elements described by truth tables contain blank cells (that is, cells that are not part of the default row).

**Erroneous Elements:**

Subroutines and actions

# (C4118) Elements containing a truth table with no inputs

**Error In:**

Completeness

**Description:**

Elements contain a truth table with no input column(s).

**Erroneous Elements:**

Elements containing truth tables

# (C4119) Activities with attribute TRANSPARENT_LEVEL set to ON which are not instances of an off-page activity chart

**Error In:**

Completeness

**Description:**

Activities contain an attribute named `TRANSPARENT_LEVEL` that is set to `ON` when there are not instances of an off-page activity chart.

> **Note**
>
> Check Model does not report on top-level activities of transparent activities.

**Example:**

The following chart contains an activity `A3`, which has an attribute called `TRANSPARENT_LEVEL` set to ON. However, `A3` is not decomposed onto an off-page chart (no @ symbol); therefore, the warning is generated.

**Erroneous Elements:**

Activities
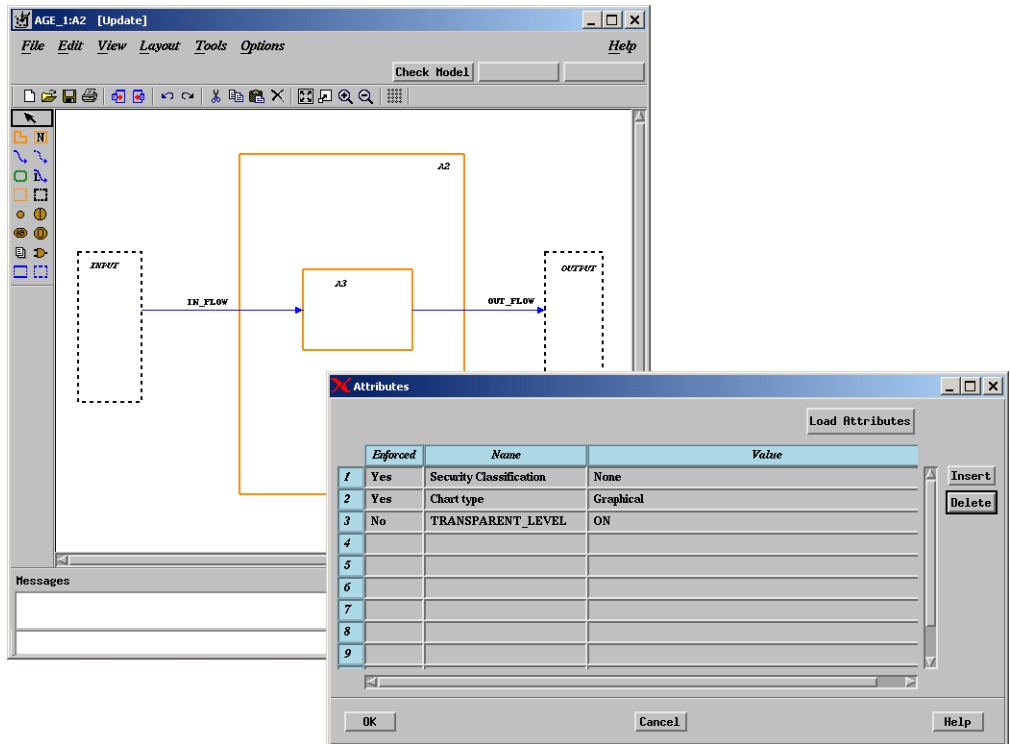
# (C4121) Continuous instances with Algebraic loops

**Error In:**

Completeness

**Description:**

The logic in a continuous diagram contains an algebraic loop, which makes it impossible to calculate the output values.

**Erroneous Elements:**

Continuous diagrams

# (C4123) Actual port binding not flowing to/from continuous diagram instance

**Error In:**

Completeness

**Description:**

Textual elements, bound to the parameters of a continuous diagram, are not shown as flowing to or from the continuous instance.

**Example:**

Activity chart A2 contains an instance of a continuous diagram CALC. A data-item is bound to one of the parameters of CALC in the properties of the instance. However, x is not shown flowing to the instance of CALC in chart A2.

**Erroneous Elements:**

Textual elements

# (C4125) Elements that flow to/from continuous instance but are not used as actual parameters

**Error In:**

Completeness

**Description:**

Textual elements, flowing to or from a continuous instance, are not used as actual parameters.

**Example:**

Activity chart A2 contains an instance of a continuous diagram CALC. Data-item x is shown flowing into the instance of CALC; however, x is not bound to one of the parameters of CALC in the properties of the instance.

**Erroneous Elements:**

Textual elements

# (C4129) "When others" not last in case structure

**Error In:**

Completeness

**Description:**

In a Switch/Case expression written in the Ada style, `when_others` must be the last statement in the expression.

**Example:**

The following expression within an activity's mini-spec will generate an error:

```
ch(SWITCH_VAL)/case_ada SWITCH_VAL is
    when_ada 1      => OUT1=RED;
    when_ada 2      => OUT1=GREEN;
    when_ada others => OUT1=NONE;
    when_ada 3      => OUT1=AMBER;
end case_ada;
```

It can be corrected as follows:

```
ch(SWITCH_VAL)/case_ada SWITCH_VAL is
    when_ada 1      => OUT1=RED;
    when_ada 2      => OUT1=GREEN;
    when_ada 3      => OUT1=AMBER;
    when_ada others => OUT1=NONE;
end case_ada;
```

**Erroneous Elements:**

Mini-specs

# (C4131) Switch/Case structure missing default entry

### Error In:

Completeness

### Description:

A Switch/Case expression within an activity's mini-spec has no `default` statement.

### Example:

The following expression will generate an error:

```
ch(SWITCH_VAL)/switch_c SWITCH_VAL {
    case_c 1: OUT1=RED;break;
    case_c 2: OUT1=GREEN;break;
    case_c 3: OUT1=AMBER;break;
}
```

It can be corrected as follows:

```
ch(SWITCH_VAL)/switch_c SWITCH_VAL {
    case_c 1: OUT1=RED;break;
    case_c 2: OUT1=GREEN;break;
    case_c 3: OUT1=AMBER;break;
    default: OUT1:NONE;
}
```

### Erroneous Elements:

Mini-specs

# (C4132) Elements input to router that are not output

**Error In:**

Completeness

**Description:**

Textual elements (data-items, conditions, and events) that are shown flowing into a router on an activity chart are not shown flowing out.

**Example:**

In the figure, data-item D is shown as an input to ROUTER_1, but is not shown flowing out of ROUTER_1.
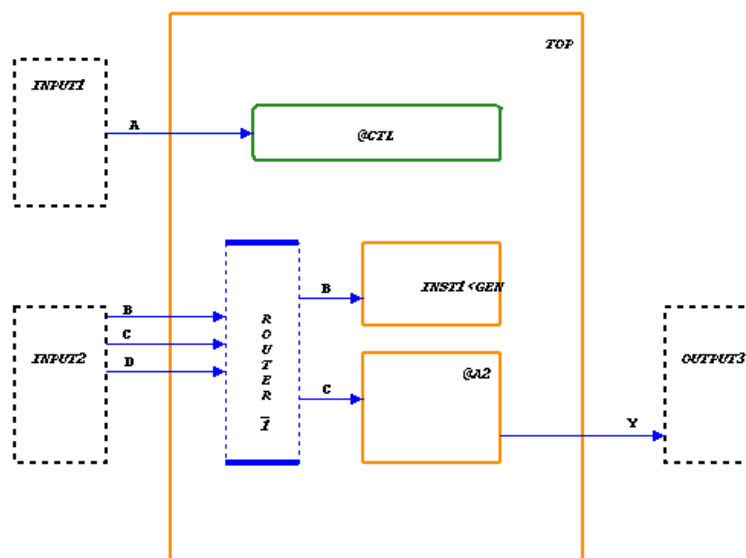


**Erroneous Elements:**

Textual elements

# (C4133) Elements output from router that are not input

### Error In:

Completeness

### Description:

Textual elements (data-items, conditions, and events) that are shown flowing out of a router on an activity chart are not shown flowing in.

### Example:

In the figure, data-item `D` is shown as an output of `ROUTER_1`, but is not shown flowing into `ROUTER_1`.
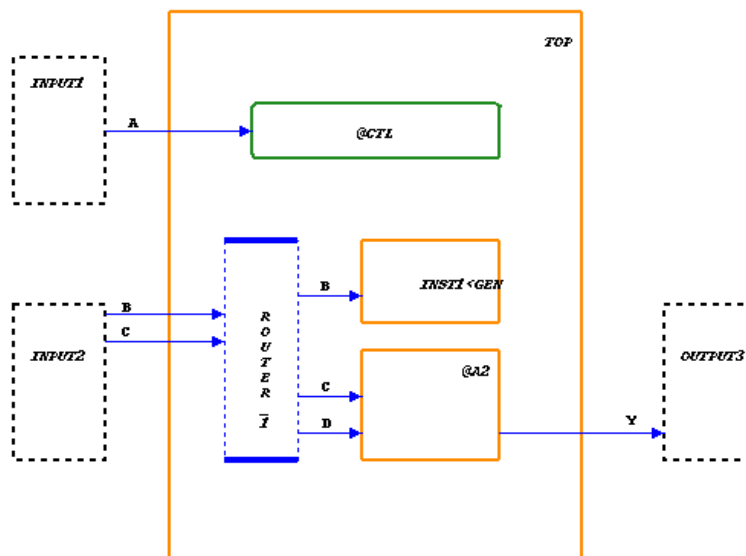


### Erroneous Elements:

Textual elements

# (C4135) Elements that flow from/to router without a matching flow in descendant chart

**Error In:**

Completeness

**Description:**

There are elements that flow from or to the router without matching flow-lines in descendant charts lower in the hierarchy.

Note that this test avoids warnings for empty information-flows. (This behavior is consistent with Check-Model tests C4058 and C4059.)

**Example:**

Consider the example in the following figure - Tested Activity-chart: TOP1

(C4135) Elements that flow from/to router without a matching flow in descendent chart:

Y flows to @AC_2 through ROUTER_1 from AC3

W flows from @AC_2 through ROUTER_1 to AC3
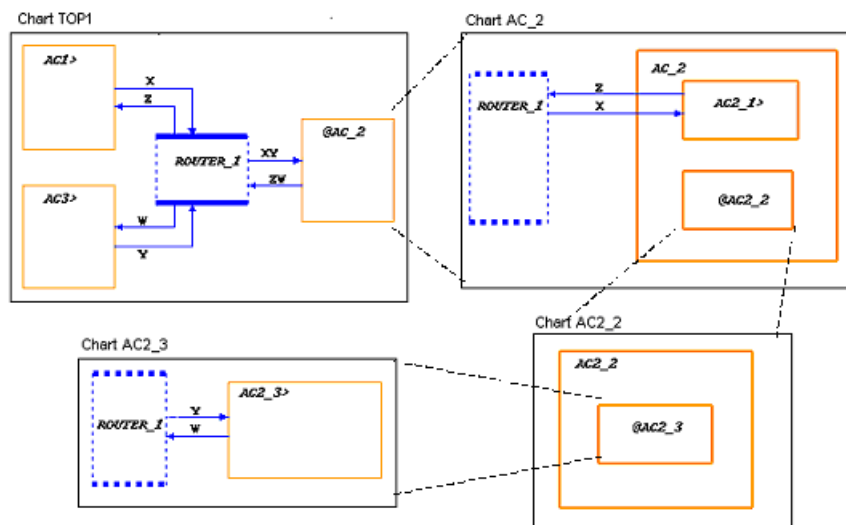


**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C4137) Elements that flow from/to router without a matching flow in ancestor chart

**Error In:**

Completeness

**Description:**

There are elements that flow from or to the router without matching flow-lines higher in the hierarchy.

Note that this test avoids warnings for empty information-flows. (This behavior is consistent with Check-Model tests C4058 and C4059.)

**Example:**

Consider the example in the following figure - Tested Activity-chart:  AC2_3

(C4137) Elements that flow from/to router without a matching flow in ancestor chart:

      Y flows from ROUTER_1 to AC2_3

      W flows from AC2_3 to ROUTER_1
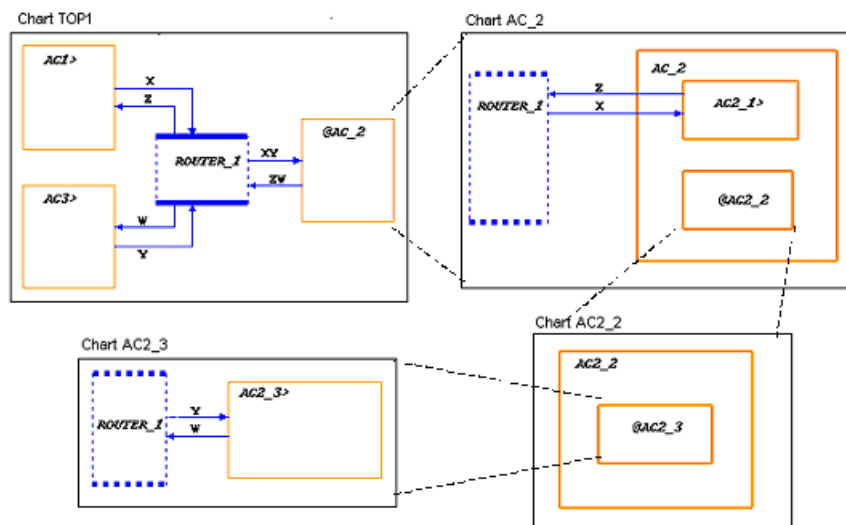


**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C4139) Elements that flow from/to instance without a matching flow in descendant chart

### Error In:

Completeness

### Description:

There are elements that flow from or to the instance activity without matching flow-lines to or from an external activity in a descendant chart.

This test is similar to C4058, but is reported when working in "Non-Strict External-Activity Resolution" mode, or when the regarded flow-lines pass through routers.

### Erroneous Elements:

Events, conditions, data-items, information-flows, and fields

# (C4141) Elements that flow from/to definition chart without a matching flow in ancestor chart

### Error In:

Completeness

### Description:

There are elements that flow to or from an external activity in the definition chart without matching flow-lines in an ancestor chart higher in the hierarchy.

This test is similar to C4059, but is reported when working in "Non-Strict External-Activity Resolution" mode, or when the regarded flow-lines pass through routers.

### Erroneous Elements:

Events, conditions, data-items, information-flows, and fields

# (C4143) Elements flowing to router from basic activity without flowing from the router to another target

**Error In:**

Completeness

**Description:**

This test warns about elements that flow to a router from a basic activity and the only possible target is the same basic activity (flow-line loopback).

**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C4145) Elements flowing from router to basic activity without flowing to the router from another source

**Error In:**

Completeness

**Description:**

This test warns about elements that flow from a Router to a basic activity and the only possible source is the same basic activity (flow-line loopback).

**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C13000) Events marked as 'Task Event' are not connected with EXTENDED Task

**Error In:**

Design

**Description:**

There are Events, which their "Its Task" Design-Attribute is not referring to a Task of type "Extended".

**Erroneous Elements:**

Events

# (C13001) Multiple tasks with the same priority

**Error In:**

Design

**Description:**

There is more than one Activity defined as "Task" in its Design-Attributes, and that have the same value in their "Priority" Design-Attribute.

**Erroneous Elements:**

Activities

# (C13002) Nested tasks

**Error In:**

Design

**Description:**

There is an Activity defined as "Task" in its Design-Attributes, and that have a Parent Activity that is defined as "Task" too.

**Erroneous Elements:**

Activities

# (C13003) Event(s), 'Its Task' is defined as default

**Error In:**

Design

**Description:**

There are Events, which their "Its Task" Design-Attribute is set to "default".

**Erroneous Elements:**

Events

# (C13004) Either 'send' or 'receive' are operated on non-message elements

**Error In:**

Design

**Description:**

There are Data-Items, which are used with the operators send (sn!()) or receive (rc!()), without being defined as Messages in their Design-Attributes.

**Erroneous Elements:**

Data-Items

# (C13005) Either 'get' or 'release' are operated on non-resource elements

**Error In:**

Design

**Description:**

There are Conditions, which are used with the operators get (gt!()) or release (rl!()), without being defined as Resources in their Design-Attributes.

**Erroneous Elements:**

Conditions

# (C13006) Data-items or/and conditions, mapping to external symbol, which is not unique

**Error In:**

Design

**Description:**

There is more than one Element with the same value to their "Bind Symbol Name" Design-Attribute.

**Erroneous Elements:**

Data-Items and Conditions

# (C13007) Operands in expressions, defined neither as double-buffered nor as having buffered access mode

**Error In:**

Design

**Description:**

There are Elements that are used with the operators changed (ch()), became-true (tr()) or became-false (fs()), and are not defined as "Double Buffered" or "Buffered Access" in their Design-Attributes.

**Erroneous Elements:**

Data-Items and Conditions

# (C13008) Multiple events, that are 'Task Event', with the same 'Its Task' and the same 'Event Mask'

**Error In:**

Design

**Description:**

There is more than one Event that are defined as "Task Event" in their Design-Attributes, and have the same value in their "Event Mask" Design-Attribute.

**Erroneous Elements:**

Events

# (C13009) Events that their its task is not appropriate

**Error In:**

Design

**Description:**

There are Events, which their "Its Task" Design-Attribute is set to an invalid "Task" name.

**Erroneous Elements:**

Events

# (C13010) Derived events that their task is not appropriate

**Error In:**

Design

**Description:**

There are Events that their "Its-Task" Design-Attribute value is referring to a Task that is not any ancestor Activity.

**Erroneous Elements:**

Data-Items and Conditions

# (C13011) Elements unnecessary defined as double-buffered

### Error In:

Design

### Description:

There are Elements that defined as "Double Buffered" in their Design-Attributes unnecessarily.

### Example:

Data-Item DI1 has its Design-Attribute "Double Buffered" set to "yes", but it is not used with any of the operators tr(), fs() or ch().

### Erroneous Elements:

Data-Items and Conditions

# (C14002) Functions implemented with incompatible language

### Error In:

Compatibility between MicroC and Rational Statemate

### Description:

A subroutine is implemented using a language such as Ada, that is not supported by the MicroC Code Generator.

### Erroneous Elements:

Subroutine

# (C14006) Usage of incompatible operators/functions

### Error In:

Unsupported functions in MicroC Code Generation

### Description:

An activity chart is using a built-in operator or function that is not supported in the MicroC Code Generator.

### Erroneous Elements:

Actions, activities, conditions, data-items, and subroutines

# (C14007) Usage of incompatible operators on a record data-item

### Error In:

Unsupported operators in MicroC Code Generation

### Description:

An activity chart is using a built-in operator on a data-item of type record that is not supported by the MicroC code generator.

### Erroneous Elements:

Actions, activities, conditions, data-items, and subroutines

# Module Chart Checks

This section documents the Check Model checks for module charts. The checks are listed in numerical order.

## (C5001) Loops in element definition

### Error In:

Correctness

### Description:

Textual elements use themselves in their own definition, via the "chain" definitions.

### Example:

```
Compound event E1 is defined as 'E2 or E3'
E2 is defined as 'E1 or E4'
```

### Erroneous Elements:

All textual elements in the loop

# (C5003) Aliases defined as non variable or non bit-array

**Error In:**

Correctness

**Description:**

Aliases are illegally defined because the element appearing in the definition of the alias is defined as a non-variable compound or constant, or it is not a bit-array.

**Erroneous Elements:**

Data-items

# (C5005) Arrays with an out of range index

**Error In:**

Correctness

**Description:**

Arrays or bit-arrays are used with an index value that is out of the defined index range.

**Example:**

```
A is an integer array from 1 to 100
There is an action in the model: \A(101):=1
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C5007) Arrays with indices in reverse order

**Error In:**

Correctness

**Description:**

There are arrays or bit-arrays for which there are references to slices whose borders are in a reverse order relative to their type definition.

**Example:**

```
A is a bit-array from 31 down to 0
There is an action in the model:
A(0.. 7):=3
```

**Erroneous Elements:**

Data-items, fields, conditions, and events

# (C5009) Constants with inconsistent definition

**Error In:**

Correctness

**Description:**

There are either constant data-items or conditions whose values do not match their type definition; constant strings, arrays, or integers whose length does not match their defined length; or a constant integer whose value is not in the defined value range.

**Erroneous Elements:**

Data-items and conditions

# (C5010) Data elements defined as arrays of bit

**Error In:**

Correctness

**Description:**

Elements are defined as array of a user-defined type (UDT), which is defined as bit.

**Example:**

```
DI1 defined as array of BIT1, which is defined as bit
```

**Erroneous Elements:**

Data-items, user-defined types, and fields

# (C5011) Elements using uninitialized context variables

**Error In:**

Correctness

**Description:**

Elements contain action expressions that use an uninitialized context variable.

**Example:**

There is a transition labeled as follows:

```
E / Y:=$X; $X:=Z
```

$X is used before any value is assigned to it.

**Erroneous Elements:**

Actions

# (C5013) Elements containing infinite While Loops

**Error In:**

Correctness

**Description:**

Elements contain action expressions in which there are infinite `while` loop statements. This check detects cases in which the `while` loop control expression does not contain a context variable and there is no `break` statement in the loop.

**Erroneous Elements:**

Actions

# (C5015) Elements containing For Loops with erroneous range indices

**Error In:**

Correctness

**Description:**

Elements contain action expressions in which there are `for` loops statements with invalid ranges.

**Example:**

There is an action that contains the following `for` loop statements:

```
for $I in 10 to 1 loop
...
for $I in 1 down to 10 loop
...
```

**Erroneous Elements:**

Actions

# (C5017) Numeric expressions using bit-arrays longer than 32 bits

**Error In:**

Correctness

**Description:**

There are elements containing numeric expressions that use bit-arrays longer than 32 bits. There is an automatic conversion from a bit-array to a numeric value only for bit-array with less than or equal to 32 bits.

**Erroneous Elements:**

Actions

# (C5019) Expressions containing operation between arrays with unequal length

**Error In:**

Correctness

**Description:**

Elements contain assignments or comparisons between arrays or array slices of different lengths.

**Example:**

An action that contains the assignment A=B, where A is defined as an array from 1 to 3, and B is an array from 1 to 5.

**Erroneous Elements:**

Events, conditions, and actions

# (C5021) Loops in index definition

### Error In:

Correctness

### Description:

Data elements use constants in their definition in a "circular" manner.

### Example:

AR is an array from 1 to N, while N is defined as rindex(AR).

### Erroneous Elements:

Data-items, conditions, events, and user-defined types

# (C5023) Defined data elements that are used incorrectly

### Error In:

Correctness

### Description:

Elements contain expressions in which data-items that are explicitly defined to be of some data-type are used in a way that does not match the type consistency and conversion rules.

### Example:

I is defined as an integer; S is defined as a string. There is an action that contains the following assignment:

```
I=S
```

### Erroneous Elements:

Events, conditions, data-items, and actions

# (C5025) Data elements without type definition that are used inconsistently

**Error In:**

Correctness

**Description:**

There are unresolved or explicitly defined data-items with no type definition that are used in expressions in an inconsistent manner.

**Example:**

D is an unresolved element (it is not explicitly defined); I is an integer; S is a string. D appears in the following assignments:

```
D=I; D=S
```

**Erroneous Elements:**

Data-items, conditions, and user-defined types

# (C5027) Predefined Statemate functions that are used incorrectly

**Error In:**

Correctness

**Description:**

Predefined Rational Statemate functions are used with parameters of a wrong type, the wrong number of parameters, or the wrong return type.

**Erroneous Elements:**

Functions

# (C5029) Subroutines that are used incorrectly

**Error In:**

Correctness

**Description:**

User functions are used inconsistently in different expressions (for example, with different return type, different parameter type, or different number of parameters).

**Erroneous Elements:**

Functions

# (C5031) Loops in compound flow-lines

**Error In:**

Correctness

**Description:**

A group of flow-line segments, linked through connectors, create a loop or cycle that does not contain a module.

**Erroneous Elements:**

Flow-line segments in the loops

# (C5033) Compound flow-lines with identical sources and targets

**Error In:**

Correctness

**Description:**

There are compound flow-lines, each of whose source set and target set contain the same basic module.

**Erroneous Elements:**

Compound flow-lines

# (C5035) Flow-line segments with illegal joints

**Error In:**

Correctness

**Description:**

Compound flow-lines are constructed by using joint connectors in a confusing order.

**Erroneous Elements:**

Components of the erroneous flow-lines

# (C5037) Flow-lines between storage modules

**Error In:**

Correctness

**Description:**

Compound flow-lines denote a flow of information between storage modules, which are passive elements.

**Erroneous Elements:**

Components of the erroneous flow-lines.

# (C5039) Flow-lines between environment modules

**Error In:**

Correctness

**Description:**

There are compound flow-lines between external modules. These flow-lines, even if they exist, should not be specified in the chart where the source and target modules are external and, therefore, out of the description scope.

**Erroneous Elements:**

Components of the erroneous flow-lines

# (C5041) Conflicting labels on flow-line components

**Error In:**

Correctness

**Description:**

There are compound flow-lines whose segments are labeled with conflicting labels.

**Erroneous Elements:**

Components of the erroneous flow-lines

# (C5043) Composition connectors with multiple entrances and multiple exits

**Error In:**

Correctness

**Description:**

There are composition connectors with multiple entering and exiting flow-lines.

**Erroneous Elements:**

Connectors

# (C5045) Instance off-page connectors with same name as in-page connector

**Error In:**

Correctness

**Description:**

Off-page diagram connectors drawn in an instance module have the same name as another in-page connector in the chart.

**Erroneous Elements**

: Instance connectors

# (C5047) Multiple instance off-page connectors with conflicting arrow directions

**Error In:**

Correctness

**Description:**

Off-page diagram connectors are drawn in an instance module in which another connector having the same name, but conflicting arrow directions, is drawn.

**Erroneous Elements:**

Instance connectors

# (C5049) Definition off-page connectors with same name as in-page connector

**Error In:**

Correctness

**Description:**

Definition off-page connectors match to an instance off-page connector and have the same name as an in-page diagram connector.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

**Erroneous Elements:**

Connectors

# (C5051) Definition off-page connectors with conflicting arrow directions in instance off-page connector

**Error In:**

Correctness

**Description:**

There are definition off-page connectors whose matching connector in the instance chart has a conflicting arrow direction.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

**Erroneous Elements:**

Definition off-page connectors

# (C5053) Elements that flow to/from instance with contradicting source/target name in definition chart

**Error In:**

Correctness

**Description:**

There are elements that flow to or from an instance module and an external module in the off-page definition chart, but the source or target name in the off-page definition chart is not the same as the name or synonym in the instance chart.

The test is performed only when "with definition charts" is specified. The error is reported in the instance chart.

**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C5055) Elements that flow to/from definition with contradicting source/target name in instance chart

**Error In:**

Correctness

**Description:**

There are elements that flow to or from an external module in the off-page definition chart and the instance module, but the source or target name in the instance chart is not the same as the name or synonym in the off-page definition chart.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C5057) Instance modules with contradicting information in definition module

### Error In:

Correctness

### Description:

There are instance modules whose type or describing activity chart is different from the type or describing activity chart entered in the properties of the top-level module in the off-page definition chart.

This test is performed only when "with definition charts" is specified. The error is reported in the instance chart.

### Erroneous Elements:

Instance modules

# (C5059) Generic instances with inconsistent parameter binding

### Error In:

Correctness

### Description:

There is inconsistency between the actual bindings in the instance box and the defined formal parameters in the generic chart.

### Erroneous Elements:

Instance boxes

# (C5061) Elements using field references that do not appear in the record / union definition

**Error In:**

Correctness

**Description:**

Elements contain expressions in which a field of an explicitly defined record or union is used, but the field does not appear in the record or union definition.

**Erroneous Elements:**

Textual elements

# (C5062) Elements erroneously using field references

**Error In:**

Completeness

**Description:**

Elements erroneously reference fields.

**Erroneous Elements:**

Any element referencing fields.

# (C5063) Formal parameters whose data-type is not global

**Error In:**

Correctness

**Description:**

Formal parameters are defined to be a particular user-defined type (UDT), but the UDT is defined in the generic chart. Because the UDT is not visible outside the generic chart, an actual binding cannot have this type.

**Erroneous Elements:**

Data-items of formal parameters

# (C5065) Actual binding whose data-type is not global

**Error In:**

Correctness

**Description:**

Data-items, used as actual bindings in an instance box, are defined to be a particular user-defined type (UDT), but the UDT is not global (the UDT is defined in a chart, not in a GDS). Because this UDT is not visible to the generic chart, a formal parameter cannot have this type.

**Erroneous Elements:**

Data-items

# (C5067) Empty Records/Unions

### Error In:

Correctness

### Description:

Data-items or data-types are defined to be a record or union, but there are no fields defined in the structure. This is a correctness error because it is a fatal error situation for some tools.

### Erroneous Elements:

Data-items and user-defined types

# (C5069) Sub-types that conflict with enumerated type definition

### Error In:

Correctness

### Description:

One of the following situations:

* Values do not appear in the definition of the enumerated type.
* Values in the subtype definition are not in the same order as in the enumerated type definition.
* Values are missing from subtype definition.

### Format:

```
<subtype name> (sub range of <enumerated-type name>):
<enum-value> - missing in <enumerated-type name>
<enum-value> - missing in <subtype name>
<enum-value> - incorrect ordinal in <subtype name>
```

**Example:**

```
WORKING_DAYS (sub range of DAYS)
MONDAY - missing in DAYS
TUE - missing in WORKING_DAYS
WED - incorrect ordinal in WORKING_DAYS
```

**Erroneous Elements:**

User-defined types

# (C5071) Tasks/Procedures that are used incorrectly

**Error In:**

Correctness

**Description:**

Tasks can be bound only to activities; any other usage, such as expressions, is erroneous. Procedures and functions cannot be bound to activities.

**Format:**

```
<subroutine name> (defined as <task/procedure/function>)
used as <procedure/function/task> in definition of <element name>
```

**Example:**

```
TASK PRINTER is used as procedure/function in definition of SCH:AN1
```

**Erroneous Elements:**

Subroutines

# (C5075) Elements containing out of range assignments of enumerated values

### Error In:

Correctness

### Description:

An element of some subtype is assigned a value that belongs to the enumerated type, but is not in the range of the subtype.

### Example:

```
In definition of SCH_ADA:ACTION WORKING_DAY:=SAT
```

### Erroneous Elements:

Textual elements, boxes, and arrows

# (C5077) Procedures/functions that are used incorrectly

### Error In:

Correctness

### Description:

Procedures are used as functions and treated in some expression as if they return a value, or functions are used as procedures and. called in an expression where an action is expected.

### Format:

```
<Function/Procedure> <subroutine name>
used as <Procedure/Function> in definition of
<element name>:<expression of subroutine call>
```

**Example:**

```
Function FACTORIAL
used as Procedure in definition of
SCH_ADA:ACTION: FACTORIAL(Y)
```

**Erroneous Elements:**

Subroutines

# (C5079) Callback procedures that are used/defined incorrectly

**Error In:**

Correctness

**Description:**

Procedures are used as callbacks, whose first parameter is either not of type string or of mode Input.

**Erroneous Elements:**

Element

# (C5081) Elements containing subroutine calls with conflicting length of parameters

### Error In:

Correctness

### Description:

There are subroutine calls where the actual parameter's length conflicts with the corresponding formal parameter definition, or whose length is unknown.

### Example:

```
STATE1
CALLBACK(A,ARRAY,3)
ARRAY - Unknown length
```

### Erroneous Elements:

Textual elements, boxes, and arrows

# (C5083) Subroutines using recursive calls

### Error In:

Correctness

### Description:

Subroutines use recursive calls either directly or by other subroutines.

### Example:

```
SR3 called from SR2
SR2 called from SR3
```

### Erroneous Elements:

Subroutines

# (C5085) Subroutines using element not declared as parameters, local variables or globals

**Error In:**

Correctness

**Description:**

Subroutines use elements not declared as parameters, local variables, or globals. All elements used in the subroutine code should be declared, including constant data-items and conditions, enumerated values, other subroutines, and user-defined types.

**Format:**

```
subroutine <subroutine name><used element name>
```

**Example:**

```
subroutine F L2
```

**Erroneous Elements:**

Subroutines

# (C5087) Subroutines that contain references to activities

### Error In:

Correctness

### Description:

Subroutines contain references to activities. This occurs only when using a global variable, which contains a reference to an activity.

### Example:

```
subroutine F
Activity A used in GLOBAL1
```

### Erroneous Elements:

Subroutines

# (C5089) Subroutines with illegal parameter type or return type

### Error In:

Correctness

### Description:

There is a subroutine where a User-Defined Type (UDT) is the return type, or the type of a parameter. The following table below lists the illegal definitions for UDTs.

| Type | Return Type | Parameter |
| --- | --- | --- |
| UDT array of string | (*) | **Illegal** |
| UDT array of bit-array | (*) | **Illegal** |
| UDT array of direct R/U | (*) | **Illegal** |
| UDT queue of anything | **Illegal** | **Illegal** |
| Array of queue (any dimension) | **Illegal** | **Illegal** |

(*) In these cases, a middle UDT is needed. For example, if the return type of a function should be an array of string, it should be defined as:

```
UDT array of STR_UDT, and STR_UDT, a string
```

**Example:**

```
subroutine F1
return type ARR_STR1, defined as array of string STR_QUEUE1,
type of parameter P1, defined as queue
```

**Erroneous Elements:**

Subroutines

# (C5091) Subroutines with illegal local variables type

**Error In:**

Correctness

**Description:**

Local variables are defined as a user-defined type (UDT), as a queue or an array (of any dimension) of a queue.

**Example:**

```
subroutine F1
STR_QUEUE1, type of local variable LV1,
defined as queue
```

**Erroneous Elements:**

Subroutines

# (C5093) Subroutines with illegal/missing RETURN command

**Error In:**

Correctness

**Description:**

There are functions without a `RETURN` command, procedures with a `RETURN` command, or a function with return type that conflicts with the returned value's type.

**Erroneous Elements:**

Subroutines

# (C5095) Subroutines that call other subroutines whose global elements are missing from the calling subroutine definition

**Error In:**

Correctness

**Description:**

Global elements of a subroutine, `SUB`, must be declared as global elements of any other subroutine calling `SUB`.

**Erroneous Elements:**

Subroutines

# (C5097) Subroutines with local variables/parameters overriding parameters/globals

**Error In:**

Correctness

**Description:**

There are subroutines with local variables that override parameters or globals, or there are parameters that override globals.

**Erroneous Elements:**

Subroutines

# (C5099) Subroutines with illegal combination of type and implementation

**Error In:**

Correctness

**Description:**

There are subroutines with an illegal combination of type and implementation.

**Erroneous Elements:**

Subroutines

# (C5101) Elements containing a truth table with illegal factorization structure

**Error In:**

Correctness

**Description:**

United input cells are not nested in a previous union; or, there are two or more rows that share the same input section, but have different output sections.

**Erroneous Elements:**

Subroutines and actions

# (C5103) Elements containing a truth table with illegal combination of inputs

**Error In:**

Correctness

**Description:**

Elements are described by truth tables using the same element twice as an input (via aliases). This check ensures that the same element is not used twice as a whole, but does not compare overlapping of slices.

**Erroneous Elements:**

Subroutines and actions

# (C5105) Elements containing a truth table with illegal combination of outputs

**Error In:**

Correctness

**Description:**

Elements are described by truth tables using the same element twice as an output (via aliases).

**Erroneous Elements:**

Subroutines and actions

# (C5107) Elements containing out of range integer/real assignments

**Error In:**

Correctness

**Description:**

Elements contain assignments of integers, where an out-of-range value is assigned to a restricted range integer, or one restricted range integer is assigned to another where there is no intersection of ranges. It tests if an integer or real value is compared with a value outside its range, as well as values assigned to integer or real variable.

**Erroneous Elements:**

States, actions, data-items, and transitions

# (C5109) Inconsistent initial value

### Error In:

Correctness

### Description:

There is a Data-Item, with one of the following data in its definition:

- ◆ Its "Min" value is smaller than its "Default Value" value.
- ◆ Its "Max" value is greater than its "Default Value" value.
- ◆ Its "Default Value" needs more bits than defined in the "Bits#" field.

### Erroneous Elements:

Data-Items

# (C5111) Inconsistent range definition

### Error In:

Correctness

### Description:

This test warns about user-defined type definitions where the sub-type ranges are inconsistent with the parent type definition.

### Example:

INT_TYPE is an integer User Defined Type and has a defined minimum/maximum range of 10 - 100 (as shown in the first figure below). SUB_INT_TYPE is a User Defined Type of INT_TYPE and has a range defined outside of 10 - 100 (as shown in the second figure).
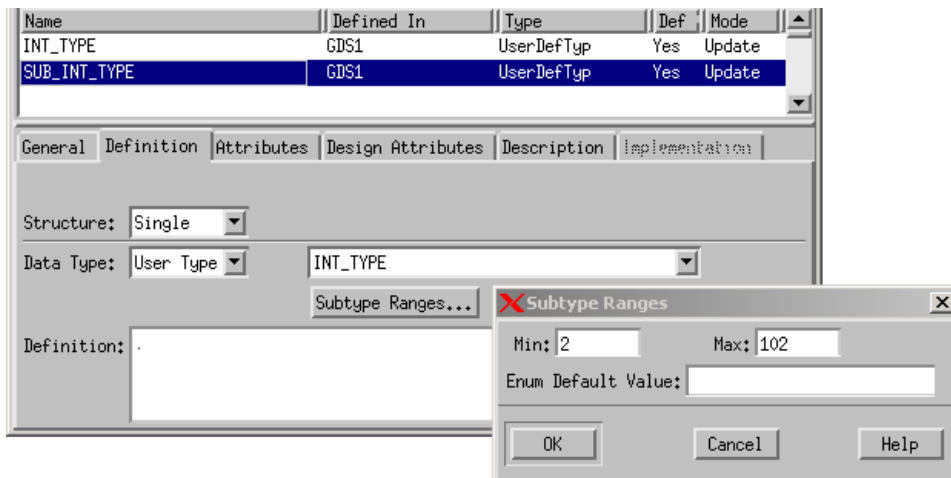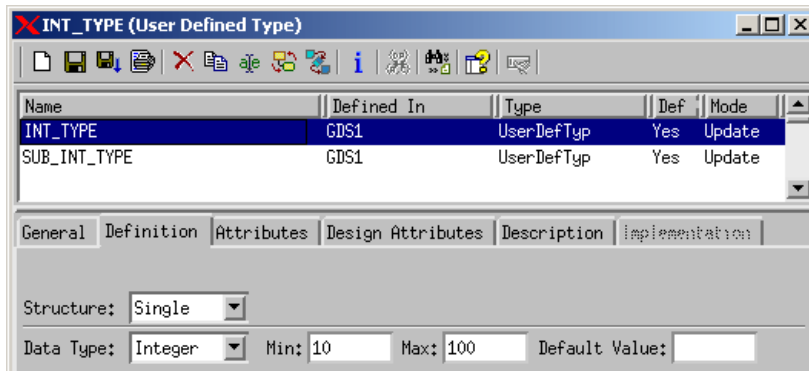
**Erroneous Elements:**

User-defined types

# (C5113) Empty information-flows

**Error In:**

Correctness

**Description:**

Array index expression does not result in an integer or contain elements with illegal type.

**Example:**

**Case I:**

Arr2 is an integer array from LeftInd2 to RightInd2.

LeftInd2 is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef2

SubDef2 is defined as:

Single None Constant with definition set to 2.34

**Case II:**

Arr1 is an integer array from LeftInd to RightInd.

LeftInd is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef

SubDef is defined as:

Single String Constant with definition set to 'My Definition'

# (C6001) Empty information-flows

**Error In:**

Completeness

**Description:**

There are information-flows whose Consists of field in the properties is empty.

**Erroneous Elements:**

Information-flows

# (C6003) Missing actual values for parameters

**Error In:**

Completeness

**Description:**

There is an instance box in whose properties there are formal parameters without actual bindings.

**Erroneous Elements:**

Instance boxes

# (C6005) Elements with incorrect or incomplete attribute definitions

**Error In:**

Completeness

**Description:**

There are elements in whose properties there is an attribute name without a corresponding attribute value, or vice-versa.

**Erroneous Elements:**

Elements of all types

# (C6006) Elements with non-unique attribute definitions

**Error In:**

Completeness

**Description:**

Element contains more than one attribute with the same name.

**Erroneous Elements:**

Any element with attributes

# (C6007) Elements that are never referenced

**Error In:**

Completeness

**Description:**

Textual elements are explicitly defined in the properties, but are not referenced in the model.

**Erroneous Elements:**

Textual elements

# (C6009) Elements using redundant context variables

**Error In:**

Completeness

**Description:**

Elements contain actions with context variables that are assigned a value, but are not used in the action.

**Erroneous Elements:**

Actions

# (C6011) Elements that could be defined lower in the hierarchy of charts

**Error In:**

Completeness

**Description:**

There are textual elements explicitly defined in the tested chart that are used only in other charts, which could be defined in the lower chart hierarchy without changing the resolution results.

The lowest chart in which they could be defined is the lowest common ancestor of all the charts in which references to this element exist.

**Erroneous Elements:**

Textual elements

# (C6013) Elements containing references to components of non-array elements

**Error In:**

Completeness

**Description:**

Elements contain data-item expressions with indices, but the accessed data-item is not an array.

**Erroneous Elements:**

Actions, conditions, and events

# (C6015) Elements containing comparisons of unions: (not supported in Code Generation)

**Error In:**

Completeness

**Description:**

Elements contain comparisons of two union data-items. The comparisons of two unions is ambiguous because the result depends on which field of the unions is the current field in each element.

**Erroneous Elements:**

Actions, conditions, and events

# (C6017) Unlabeled compound flow-lines

**Error In:**

Completeness

**Description:**

There are compound flow-lines, all of whose components are unlabeled.

**Erroneous Elements:**

Components of erroneous flow-lines

# (C6019) Unlabeled flow-lines entering or exiting composition connectors

**Error In:**

Completeness

**Description:**

There are flow-line segments whose source or target is a composition connector and to which no label is attached.

**Erroneous Elements:**

Unlabeled flow-line segments

# (C6021) Flow-lines with incomplete source definition

**Error In:**

Completeness

**Description:**

There are flow-lines whose source is not connected to a module or connector. The source of such flow-lines appears in the module chart as a small line perpendicular to the flow-line.

**Erroneous Elements:**

Flow-lines

# (C6023) Flow-lines with incomplete target definition

**Error In:**

Completeness

**Description:**

There are flow-lines whose target is not connected to a module or connector. The target of such flow-lines appears in the activity chart as a small line perpendicular to the flow-line.

**Erroneous Elements:**

Flow-lines

# (C6025) Compound flow-lines with identical sources

**Error In:**

Completeness

**Description:**

There are compound flow-lines whose source set contains the same module more than once, or a module and one of its descendants.

**Erroneous Elements:**

Components of the erroneous flow-lines

# (C6027) Compound flow-lines with identical targets

**Error In:**

Completeness

**Description:**

There are compound flow-lines whose target set contains the same module more than once, or a module and one of its descendants.

**Erroneous Elements:**

Components of the erroneous flow-lines

# (C6029) Composition connectors with illegally located sources or targets

**Error In:**

Completeness

**Description:**

There are composition connectors whose components flow from sources or to targets that are outside the module in which the connector resides.

**Erroneous Elements:**

Connectors

# (C6031) Isolated connectors

**Error In:**

Completeness

**Description:**

There are connectors with no entrances and no exits.

**Erroneous Elements:**

Connectors

# (C6033) Connectors with an entrance and no exit

**Error In:**

Completeness

**Description:**

There are junction or composition connectors, or a group of in-page diagram connectors with the same name, with an entrance, but no exit.

**Erroneous Elements:**

Connectors

# (C6035) Connectors with an exit and no entrance

**Error In:**

Completeness

**Description:**

There are junction or composition connectors, or a group of in-page diagram connectors, with the same name, with an exit, but no entrance.

**Erroneous Elements:**

Connectors

# (C6037) Junction or composition connectors with a single entrance and a single exit

**Error In:**

Completeness

**Description:**

There are junction or composition connectors with a single entrance and a single exit and, therefore, are probably redundant.

**Erroneous Elements:**

Connectors

# (C6039) Composition connectors with missing flow elements

### Error In:

Completeness

### Description:

There are composition connectors whose record decomposition is not complete.

### Erroneous Elements:

Data-items

# (C6041) Unresolved elements

### Error In:

Completeness

### Description:

There are reference elements for which no element with the same name is explicitly defined in the relevant scope. This check also checks resolution of external modules in an off-page chart to modules in the parent chart.

### Erroneous Elements:

All types of elements

# (C6043) The tested chart is a definition chart with multiple top-level boxes

**Error In:**

Completeness

**Description:**

The tested chart is used as an off-page definition of an instance module and has multiple top-level modules.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

**Erroneous Elements:**

Tested chart

# (C6045) The tested chart is a definition chart with empty top-level box

**Error In:**

Completeness

**Description:**

The tested charts is used as an off-page definition of an instance module and has a single, empty, top-level module.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

**Erroneous Elements:**

Tested chart

# (C6047) Instance off-page connectors without a matching definition off-page connector

**Error In:**

Completeness

**Description:**

Instance off-page connectors have no matching definition diagram connector.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

**Erroneous Elements:**

Connectors

# (C6049) Elements that flow from/to instance without a matching flow in definition chart

**Error In:**

Completeness

**Description:**

There are elements that flow from or to the instance module without matching flow-lines to or from an external module in the off-page definition chart. Note that the elements flowing on flow-lines are matched, not the sources or targets of the flow.

The test is performed only when "with definition charts" is specified. The error is reported in the instance chart.

**Erroneous Elements:**

Events, conditions, data-items, information-flows, and fields

# (C6051) Elements that flow from/to definition without a matching flow in instance chart

### Error In:

Completeness

### Description:

There are elements that flow to or from an external module in the definition chart without matching flow-lines from or to the instance. Note that the elements flowing on the flow-lines are matched, not the sources or targets of the flow.

The test is performed only when "with definition charts" is specified. The error is reported in the definition chart.

### Erroneous Elements:

Events, conditions, data-items, information-flows, and fields

# (C6053) Actual parameters not flowing to/from generic instance

### Error In:

Completeness

### Description:

Actual parameters are bound to some Input or Output formal parameter in an instance box, but are not specified as Input or Output to that box.

### Erroneous Elements:

Events, conditions, data-items, and fields

# (C6055) Elements that flow to/from generic instance but are not used as actual parameters

**Error In:**

Completeness

**Description:**

There are elements flowing into or out of an instance box of a generic chart, but they are not declared as an actual input or output parameter.

**Erroneous Elements:**

Events, conditions, data-items, and fields

# (C6057) External elements not declared as formal parameters

**Error In:**

Completeness

**Description:**

There are elements that flow from or to an external module, to or from the top-level module in a generic chart, but do not appear as In or Out formal parameters of the generic chart.

**Erroneous Elements:**

Events, conditions, and data-items

# (C6059) Formal parameters not appearing on external flow-lines

**Error In:**

Completeness

**Description:**

There are elements declared as In or Out parameters of a generic module chart, but they do not flow into or out of the top-level module to an external box in the generic chart.

**Erroneous Elements:**

Events, conditions, and data-items

# (C6061) Data elements with incomplete type definition

**Error In:**

Completeness

**Description:**

There are data elements whose type definition is missing or incomplete.

**Erroneous Elements:**

Events, conditions, data-items, and user-defined types

# (C6063) Constant parameter arrays with incomplete length definition

**Error In:**

Completeness

**Description:**

There are constant parameter arrays where either the definition of the formal parameter in the generic chart is incomplete, or they are bound to an actual parameter of undefined length.

**Erroneous Elements:**

Data-items and conditions

# (C6065) Defined elements that can be merged with elements from ancestor charts

**Error In:**

Completeness

**Description:**

There are textual elements explicitly defined in a chart that have matching, explicitly defined elements with the same name in ancestor charts. Both elements have the same type in both charts; therefore, they could probably be merged.

**Erroneous Elements:**

Textual elements

# (C6067) The tested Module chart contains no modules

### Error In:

Completeness

### Description:

The tested module chart contains no modules.

### Erroneous Elements:

Tested module chart

# (C6069) Elements overriding global definitions

### Error In:

Completeness

### Description:

There are textual elements with the same name as elements defined in a global definition set.

### Erroneous Elements:

Textual elements

# (C6071) Subroutines not implemented in any language

**Error In:**

Completeness

**Description:**

There are subroutines that have no implementation in the   Rational Statemate Action Language, Procedural Statechart/Flowchart, Ada Code, K&R C Code, ANSI C Code, and so on.

**Format:**

```
<subroutine name>
```

**Example:**

```
CALLBACK
```

**Erroneous Elements:**

Subroutines

# (C6073) Subroutine parameters of a non-unique user-defined type

**Error In:**

Completeness

**Description:**

A parameter of a subroutine is defined as a user-defined type (UDT), whose name is not unique in the model.

**Example:**

```
P2 parameter of procedure CALLBACK
Using nonunique user-defined type DAYS
```

**Erroneous Elements:**

Subroutines

# (C6075) Enumerated types with non-unique user-defined type

**Error In:**

Completeness

**Description:**

There are enumerated types with values that appear in other unrelated enumerated type definitions (not in a subtype).

**Example:**

```
user-defined type COLORS
BLUE, exists also in SCH:MOODS
```

**Erroneous Elements:**

User-defined types

# (C6076) Data-Elements with Enumerated Value names

**Error In:**

Correctness

**Description:**

Textual element has same name as enumerated values.

**Erroneous Elements:**

Textual elements

# (C6077) Textual subroutines using uninitialized local variables

## Error In:

Completeness

## Description:

Local variables of a textual subroutine are used before they are affected or assigned some value.

## Format:

```
<local variable name> in <subroutine name>
```

## Example:

```
LOCAL1 in F1
```

## Erroneous Elements:

Subroutines

# (C6079) Textual subroutine using redundant local variables

### Error In:

Completeness

### Description:

Local variables of a textual subroutine are affected, but not used.

### Format:

```
<local variable name> in <subroutine name>
```

### Example:

```
LOCAL1 in F1
```

### Erroneous Elements:

Subroutines

# (C6081) Subroutines containing wrong usage of parameters or globals

### Error In:

Completeness

### Description:

Subroutines contain the wrong usage of parameters or globals. For example:

- In parameters that are affected by the subroutine.
- In parameters that are not used by the subroutine.
- Out parameters that are used by the subroutine.
- Out parameters that are not affected by the subroutine.
- InOut parameters that are not used by the subroutine.
- InOut parameters that are not affected by the subroutine

**Example:**

```
subroutine CALLBACK
Input parameter P1 not used
```

**Erroneous Elements:**

Subroutines

# (C6083) Subroutines with incomplete type definition

**Error In:**

Completeness

**Description:**

There are subroutines with incomplete type definitions.

**Erroneous Elements:**

Subroutines

# (C6085) Local variables not used in the subroutine

**Error In:**

Completeness

**Description:**

Local variables are not used in the subroutine code.

**Example:**

```
subroutine A Local variable L3 not used
```

**Erroneous Elements:**

Subroutines

# (C6087) Elements with illegal "Selected Implementation"

**Error In:**

Completeness

**Description:**

The selected implementation is not one of the existing implementations of the element.

**Erroneous Elements:**

Subroutines

# (C6089) Elements containing a truth table with empty default row

**Error In:**

Completeness

**Descriptions:**

Elements described by truth tables contain a default row with its the output columns blank and the action section empty.

**Erroneous Elements:**

Subroutines and actions

# (C6091) Elements containing a truth table with empty cells

**Error In:**

Completeness

**Description:**

Elements described by truth tables contain blank cells (that is, cells that are not part of the default row).

**Erroneous Elements:**

Subroutines and actions

# (C6093) Elements containing a truth table with no inputs

**Error In:**

Completeness

**Description:**

Elements contain a truth table with no input column(s).

**Erroneous Elements:**

Elements containing truth tables

# Global Definition Set Checks

This section documents the Check Model checks for Global Definition Sets (GDSs). The checks are listed in numerical order.

## (C7001) Elements with incorrect or incomplete attribute definitions

### Error In:

Completeness

### Description:

There are elements whose properties contain an attribute name without a corresponding attribute value, or vice-versa.

### Erroneous Elements:

Elements of all types

## (C7002) Elements containing references to components of non-array elements

### Error In:

Completeness

### Description:

Elements contain data-item expressions with indices, but the accessed data-item is not an array.

### Erroneous Elements:

Element

# (C7003) The tested global definition set contains no elements

**Error In:**

Completeness

**Description:**

There are no elements defined in the tested global definition set (GDS).

**Erroneous Elements:**

The tested GDS

# (C7004) Elements that are not referenced in Statemate Expressions and Definitions

**Error In:**

Completeness

**Description:**

Checks for elements that are defined in GDS but are not referenced in the model.

**Erroneous Elements:**

Element

# (C7005) Unresolved elements

**Error In:**

Completeness

**Description:**

There are reference elements for which no elements with the same name are explicitly defined in the relevant scope.

**Erroneous Elements:**

All types of elements

# (C7006) Elements with non-unique attribute definitions

**Error In:**

Completeness

**Description:**

Element contains more than one attribute with the same name.

**Erroneous Elements:**

Any element with attributes

# (C7007) Data elements with incomplete type definition

**Error In:**

Completeness

**Description:**

There are data elements whose type definition is missing or incomplete.

**Erroneous Elements:**

Conditions, data-items, and user-defined types

# (C7009) Subroutines not implemented in any language

**Error In:**

Completeness

**Description:**

Subroutines have no implementation in the Rational Statemate Action Language, Procedural Statechart/Flowchart, Ada Code, K&R C Code, ANSI C Code, and so on.

**Example:**

CALLBACK

**Erroneous Elements:**

Subroutines

# (C7011) Subroutines containing parameters defined as a non-unique User-defined type

**Error In:**

Completeness

**Description:**

A parameter of a subroutine is defined a user-defined type, but its name is not unique in the model.

**Example:**

```
P2 parameter of procedure CALLBACK
Using nonunique user-defined type DAYS
```

**Erroneous Elements:**

Subroutines

# (C7013) Enumerated types with non-unique value names

**Error In:**

Completeness

**Description:**

Enumerated types with values appear in other, unrelated enumerated type definitions (not in a subtype).

**Example:**

```
user-defined type COLORS
BLUE, exists also in SCH:MOODS
```

**Erroneous Elements:**

User-defined types

# (C7014) Data-Elements with Enumerated Value names

**Error In:**

Correctness

**Description:**

Textual element has same name as enumerated values.

**Erroneous Elements:**

Textual elements

# (C7015) Textual subroutines using uninitialized local variables

**Error In:**

Completeness

**Description:**

Local variables of a textual subroutine are used before they are affected or assigned some value.

**Example:**

```
LOCAL1 in F1
```

**Erroneous Elements:**

Subroutines

# (C7017) Textual subroutines using redundant local variables

**Error In:**

Completeness

**Description:**

Local variables of a textual subroutine are affected, but not used.

**Example:**

```
LOCAL1 in F1
```

**Erroneous Elements:**

Subroutines

# (C7019) Subroutines containing wrong usage of parameters or globals

**Error In:**

Completeness

**Description:**

Subroutines contain the wrong usage of parameters or globals, such as:

- In parameters that are affected by the subroutine.
- In parameters that are not used by the subroutine.
- Out parameters that are used by the subroutine.
- Out parameters that are not affected by the subroutine
- InOut parameters that are not used by the subroutine
- InOut parameters that are not affected by the subroutine

**Example:**

```
subroutine CALLBACK
Input parameter P1 not used
```

**Erroneous Elements:**

Subroutines

# (C7020) Subroutines with incomplete type definition

**Error In:**

Completeness

**Description:**

There are subroutines with incomplete type definitions.

**Erroneous Elements:**

Subroutines

# (C7021) Local variables not used in the subroutine

**Error In:**

Completeness

**Description:**

A subroutine's selected implementation defines a local variable that is never used in the implementation.

**Erroneous Elements:**

Subroutine

# (C7022) Elements containing a truth table with no inputs

**Error In:**

Completeness

**Description:**

Elements contain a truth table with no input column(s).

**Erroneous Elements:**

Elements containing truth tables

# (C7023) Elements with illegal "Selected Implementation"

**Error In:**

Completeness

**Description:**

The selected implementation is not one of the existing implementations of the element.

**Erroneous Elements:**

Subroutines, actions, and truth tables

# (C7024) Elements containing a truth table with empty default row

**Error In:**

Completeness

**Description:**

The element is described by a truth table that contains an empty default row (the output columns are blank and the action section is empty).

**Erroneous Elements:**

Subroutines, actions, and truth tables

# (C7025) Elements containing a truth table with empty cells

**Error In:**

Completeness

**Description:**

The element is described by a truth table that contains empty cells that are not part of the default row.

**Erroneous Elements:**

Subroutines, actions, and truth tables

# (C7101) Loops in element definition

### Error In:

Correctness

### Description:

Elements use themselves in their definition, via chain definitions.

### Example:

Data-type `T1` is defined as integer with a minimum value of `K1` and a maximum value of `K2`; `K1` a constant data-item defined to be of type `T1`.

### Erroneous Elements:

All elements in the loop

# (C7103) Arrays with indices in reverse order

### Error In:

Correctness

### Description:

There are arrays or bit-arrays for which there are references to slices whose borders are in a reverse order, relative to their type definition.

### Example:

A is a bit-array from 31 down to 0. There is an action in the model:

```
A (0..7)=3
```

### Erroneous Elements:

Data-items, fields, conditions, and events

# (C7105) Constants with inconsistent definition

**Error In:**

Correctness

**Description:**

One of the following:

- There are constant data-items or conditions whose values do not match their type definition.
- There are constant strings, arrays, or integers whose length do not match their defined length.
- There are constant integers whose values are not in the defined value range.

**Erroneous Elements:**

Data-items and conditions

# (C7106) Data elements defined as arrays of bits

**Error In:**

Correctness

**Description:**

Elements are defined as an array of a user-defined type (UDT) defined as bit.

**Example:**

```
DI1 defined as array of BIT1, which is defined as bit
```

**Erroneous Elements:**

Data-items, UDTs, and fields

# (C7107) Defined data elements that are used incorrectly

**Error In:**

Correctness

**Description:**

Elements contain expressions in which data-items explicitly defined as a particular data-type are used in a way that does not match the type consistency and conversion rules.

**Example:**

I is defined as an integer; S is defined as a string. There is an action that contains the following assignment:

        I=S

**Erroneous Elements:**

States, transitions, events, conditions, data-items, and actions

# (C7109) Data elements without type definition that are used inconsistently

**Error In:**

Correctness

**Description:**

Elements contain expressions in which data-items are not explicitly defined to be of a particular data-type are used in a way that does not match the type consistency and conversion rules.

**Example:**

I is used as an integer; S is used as a string. There is an action that contains the following assignment:

```
I=S
```

**Erroneous Elements:**

States, transitions, events, conditions, data-items, and actions

# (C7110) Subroutines that are used incorrectly

**Error In:**

Correctness

**Description:**

There are subroutine calls with expressions, constants, or compound elements used as Out or InOut parameters. Only variable/alias data-items or conditions can be used as output parameters.

**Example:**

```
subroutine SR1
Output parameter P1:
assigned the expression A1+B in ACH:ACT2:
SR1(A1+B)
```

**Erroneous Elements:**

Subroutines

# (C7111) Elements having multiple definitions

**Error In:**

Correctness

**Description:**

Elements with the same name appear in multiple global definition sets.

**Erroneous Elements:**

User-defined types, data-items, and conditions

# (C7112) Elements using field references that do not appear in the record/union definition

**Error In:**

Correctness

**Description:**

There are elements that contain expressions in which a field of an explicitly defined record or union is used, but the field does not appear in the definition of the record or union.

# (C7113) Empty Records/Unions

**Error In:**

Correctness

**Description:**

Data-types are defined to be a record or union, but there are no fields defined in the structure. This is a correctness error because it is a fatal error situation for some tools.

**Erroneous Elements:**

User-defined types

# (C7114) Elements erroneously using field references

**Error In:**

Completeness

**Description:**

Elements erroneously reference fields.

**Erroneous Elements:**

Any element referencing fields.

# (C7116) Tasks/Procedures that are used incorrectly

**Error In:**

Correctness

**Description:**

Tasks can be bound to activities only. Any other usage, such as in expressions, is incorrect. Procedures and functions cannot be bound to activities.

**Example:**

TASK PRINTER is used as procedure/function in definition of SCH:AN1

**Erroneous Elements:**

Subroutines

# (C7119) Elements containing subroutine calls with conflicting length of parameters

### Error In:

Correctness

### Description:

There are subroutine calls where the actual parameter's length conflicts with the corresponding formal parameter definition, or whose length is unknown.

### Example:

```
STATE1
CALLBACK(A,ARRAY,3)
ARRAY - Unknown length
```

### Erroneous Elements:

Textual elements, boxes, and arrows

# (C7121) Subroutines using recursive calls

### Error In:

Correctness

### Description:

There are subroutines using recursive calls (either directly or by other subroutines).

### Example:

```
SR3 called from SR2
SR2 called from SR3
```

### Erroneous Elements:

Subroutines

# (C7123) Subroutines using elements not declared as parameters, local variables, or globals

**Error In:**

Correctness

**Description:**

Subroutines use elements that are not declared as parameters, local variables, or globals. All elements used in the subroutine code should be declared, including constant data-items and conditions, enumerated values, other subroutines, and user-defined types.

**Example:**

subroutine F L2

**Erroneous Elements:**

Subroutines

# (C7125) Subroutines that contain references to activities

**Error In:**

Correctness

**Description:**

Subroutines contain references to activities. This is possible only when using a global variable, which contains a reference to an activity.

**Example:**

```
subroutine F
Activity A used in GLOBAL1
```

**Erroneous Elements:**

Subroutines

# (C7127) Subroutines with illegal parameter type or return type

**Error In:**

Correctness

**Description:**

In a subroutine where a UDT is the return type or the type of a parameter, the UDT has an illegal definition. The following table lists the illegal definitions for UDTs.

| Type | Return Type | Parameter |
|---|---|---|
| **UDT array of string** | (*) | ** Illegal ** |
| **UDT array of bit-array** | (*) | ** Illegal ** |
| **UDT array of direct R/U** | (*) | ** Illegal ** |
| **UDT queue of anything** | ** Illegal ** | ** Illegal ** |
| **Array of queue (any dimension)** | ** Illegal ** | ** Illegal ** |

(*) In these cases, a middle user-defined type, is needed. For example, if the return type of a function is an array of string, it should be defined as:

```
UDT array of STR_UDT, and STR_UDT, a string
```

**Example:**

```
subroutine F1
return type ARR_STR1, defined as array of string STR_QUEUE1,
type of parameter P1, defined as queue
```

**Erroneous Elements:**

Subroutines

# (C7129) Subroutines with illegal local variables type

**Error In:**

Correctness

**Description:**

A local variable is defined as a user-defined type (UDT), which is a queue or an array (of any dimension) of a queue.

**Example:**

```
subroutine F1
STR_QUEUE1, type of local variable LV1, defined as queue
```

**Erroneous Elements:**

Subroutines

# (C7131) Subroutines with illegal/missing RETURN command

**Error In:**

Correctness

**Description:**

There are functions without a RETURN command, procedures with a RETURN command, or a function with a return type that conflicts with the returned value's type.

**Erroneous Elements:**

Subroutines

# (C7133) Subroutines with local variables/parameters overriding parameters/globals

### Error In:

Correctness

### Description:

There are subroutines with local variables that override parameters or globals, or parameters override globals.

### Erroneous Elements:

Subroutines

# (C7134) Subroutines with illegal combination of type and implementation

### Error In:

Correctness

### Description:

A subroutine has a selected implementation that is not supported for the subroutine type (task, function, or procedure).

### Erroneous Elements:

Subroutine

# (C7135) Elements containing a truth table with illegal factorization structure

**Error In:**

Correctness

**Description:**

United input cells are not nested in a previous union, or two or more rows share the same input section, but have different output sections.

**Erroneous Elements:**

Subroutines, actions, and truth tables

# (C7136) Elements containing a truth table with illegal combination of inputs

**Error In:**

Correctness

**Description:**

Elements described by truth tables use the same element twice as an input, via aliases. The check ensures that the same element is not used twice as a whole (but the check will not compare overlapping of slices).

**Erroneous Elements:**

Subroutines, actions, and truth tables

# (C7137) Elements containing a truth table with illegal combination of outputs

**Error In:**

Correctness

**Description:**

Elements described by truth tables use the same element twice as an output, via aliases.

**Erroneous Elements:**

Subroutines, actions, and truth tables

# (C7139) Elements containing out of range integer/real assignments

**Error In:**

Correctness

**Description:**

Elements contain assignments of integers, where an out-of-range value is assigned to a restricted range integer; or one restricted range integer is assigned to another, where there is no intersection of ranges. It tests if an integer or real value is compared with a value outside its range, as well as values assigned to integer or real variable.

**Erroneous Elements:**

States, actions, data-items, transitions, and subroutines.

Global elements of a subroutine, SUB, must be declared as global elements of any other subroutine calling SUB.

# (C7141) Inconsistent initial value

**Error In:**

Correctness

**Description:**

An element has a default value that is inconsistent with its type and range definition.

**Example:**

Integer data-type x has a minimum value of 0, and max value 20 and Default Value A. Integer data-item A is a constant set to 40. A is outside the Min and Max for X so it is an inconsistent initial value.

**Erroneous Elements:**

Integer and real data-items

# (C7143) Inconsistent range definition

**Error In:**

Correctness

**Description:**

This test warns about user-defined type definitions where the sub-type ranges are inconsistent with the parent type definition.

**Example:**

INT_TYPE is an integer User Defined Type and has a defined minimum/maximum range of 10 - 100 (as shown in the first figure below). SUB_INT_TYPE is a User Defined Type of INT_TYPE and has a range defined outside of 10 - 100 (as shown in the second figure below).
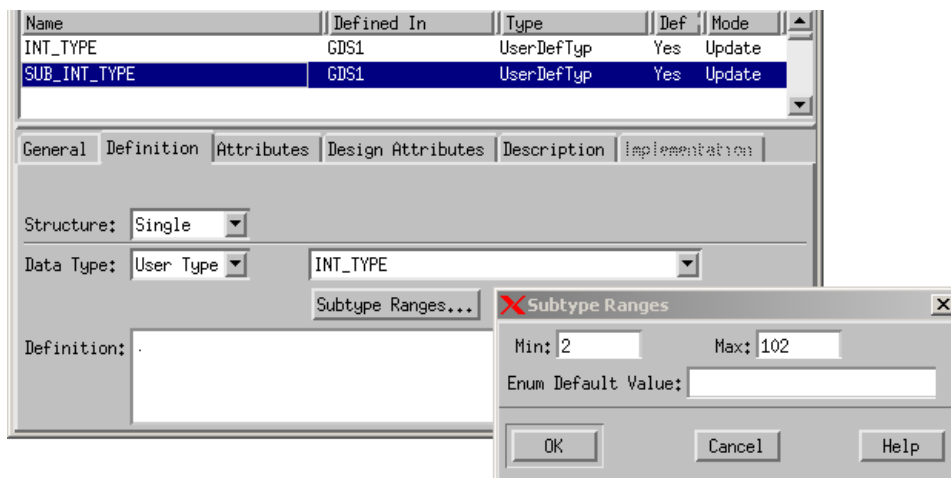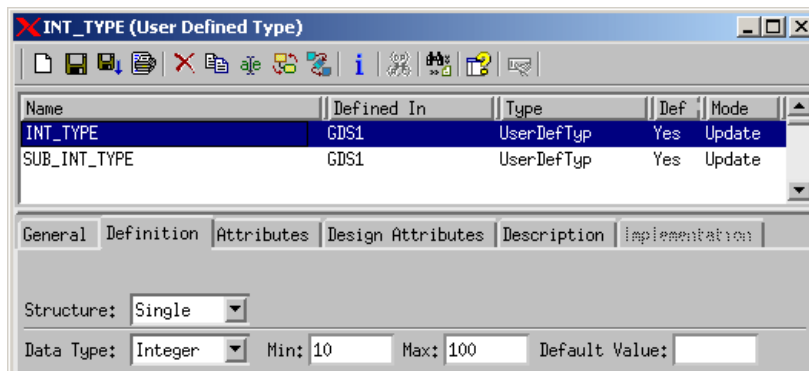
**Erroneous Elements:**

User-defined types

# (C7145) Empty information-flows

**Error In:**

Correctness

**Description:**

Array index expression does not result in an integer or contain elements with illegal type.

**Example:**

**Case I:**

Arr2 is an integer array from LeftInd2 to RightInd2.

LeftInd2 is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef2

SubDef2 is defined as:

Single None Constant with definition set to 2.34

**Case II:**

Arr1 is an integer array from LeftInd to RightInd.

LeftInd is defined as:

Single, Constant, with Data Type set to None, and definition set to SubDef

SubDef is defined as:

Single String Constant with definition set to 'My Definition'

# Sequence Diagram Checks

This section documents the Check Model checks for sequence diagrams. The checks are listed in numerical order:

## (C7801) Unlabeled message-lines

**Error In:**

Completeness

**Description:**

A message line was drawn but not labeled.

**Erroneous Elements:**

Message lines

## (C7802) Message-lines with incomplete source definition

**Error In:**

Completeness

**Description:**

The source of a message line was not drawn on a life-line. The offending message line is indicated on the sequence diagram with a small "x" at its source and target.

**Erroneous Elements:**

Message lines

# (C7803) Message-lines with incomplete target definition

**Error In:**

Completeness

**Description:**

The target of a message line was not drawn on a life-line. The offending message line is indicated on the sequence diagram with a small "x" at its source and target.

**Erroneous Elements:**

Message-lines

# (C7804) Unlabeled timing constraints

**Error In:**

Completeness

**Description:**

A timing constraint was drawn but not labeled.

**Erroneous Elements:**

Timing constraints

# (C7805) Life-Lines without any input or output

**Error In:**

Completeness

**Description:**

A life-line in a sequence diagram is neither the source nor the target for any message lines.

**Erroneous Elements:**

Life-lines

# (C7806) The tested sequence-diagram contains no life-lines

**Error In:**

Completeness

**Description:**

A sequence diagram was created, but no life-lines were drawn in the diagram.

**Erroneous Elements:**

Sequence diagrams

# (C7807) Unresolved elements

**Error In:**

Completeness

**Description:**

The sequence diagram contains references to elements that are not explicitly defined in the relevant scope. The scope for a sequence diagram is set in the chart properties for the sequence diagram.

**Erroneous Elements:**

All life-lines and all elements used in message-line labels

# Statechart and Activity Chart Relation Checks

This section documents the Check Model checks for Statechart (SC) and Activity Chart (AC) relation checks. The checks are listed in numerical order:

## (C7201) The control activity is not connected to a statechart

### Error In:

Completeness

### Description:

The control activity of the tested activity is not connected to a statechart.

### Erroneous Elements:

Control activity

## (C7203) The control activity is connected to an unresolved statechart

### Error In:

Completeness

### Description:

The control activity of the tested activity is connected to a statechart that does not exist in the workarea.

### Erroneous Elements:

Control activity

# (C7205) Unresolved elements

**Error In:**

Completeness

**Description:**

Activities referenced by the statechart are not defined in the activity controlled by the statechart.

**Erroneous Elements:**

Activities

# (C7207) Input elements not used in the statechart

**Error In:**

Completeness

**Description:**

Data elements specified in the activity chart as direct input to the control activity of the tested activity are not used inside the statechart.

**Erroneous Elements:**

Events, conditions, data-items, and information-flows

# (C7209) Output elements not affected by the statechart

**Error In:**

Completeness

**Description:**

Data elements specified in the activity chart as direct output of the control activity of the tested activity are not affected, changed, or generated inside the statechart.

**Erroneous Elements:**

Events, conditions, data-items, and information-flows

# (C7211) Elements that are used in a statechart but are neither input nor affected internally

### Error In:

Completeness

### Description:

Variable events, conditions, or data-items are used (evaluated) in the statechart, but are not specified in the activity chart as input to the control activity (directly or to one of its ancestors), nor are they generated or modified by actions in the statechart.

### Erroneous Elements:

Events, conditions, and data-items

# (C7213) Elements affected by the statechart that are neither used nor output

### Error In:

Completeness

### Description:

Elements are affected (modified or generated in actions) by the statechart, but they are neither specified as output of the control activity nor used in the statechart.

### Erroneous Elements:

Events, conditions, and data-items

# (C7215) Elements affected and used in the statechart but are neither input nor output

**Error In:**

Completeness

**Description:**

Elements are affected in actions and are used in the statechart (for example, in triggers), but are defined outside the statechart (in an ancestor activity chart) and do not flow into or from the control activity described by the statechart.

**Erroneous Elements:**

Events, conditions, and data-items

# (C7217) Subactivities not activated by the statechart

**Error In:**

Completeness

**Description:**

Subactivities of the tested activity are neither started in any action, nor performed throughout or within any state in the controlling statechart.

**Erroneous Elements:**

Activities

# (C7219) Missing termination connector in the statechart of a self-terminated activity

**Error In:**

Completeness

**Description:**

The tested activity is defined with termination type reactive self, but there is no termination connector in the statechart connected to the control activity.

**Erroneous Elements:**

Tested activity

# (C7221) Controlled-termination subactivities that are not stopped by the statechart

**Error In:**

Completeness

**Description:**

Activities with termination type reactive controlled (which are subactivities of the tested activity) are neither stopped in any action, nor performed throughout any state in the statechart connected to the control activity.

**Erroneous Elements:**

Activities

# (C7223) Procedure-like subactivities that are stopped by the statechart

**Error In:**

Completeness

**Description:**

Activities with a procedure-like termination type are stopped in actions or static reactions in the statechart connected to the control activity.

**Erroneous Elements:**

Activities

# (C7225) Activities suspended by the Statechart but not resumed, or vice versa

**Error In:**

Completeness

**Description:**

Activities are suspended by actions in the controlling statechart, but are not resumed in actions, or vice versa.

**Erroneous Elements:**

Activities

# (C7227) Empty controlling statechart

**Error In:**

Completeness

**Description:**

The statechart that describes the control activity contains no states.

**Erroneous Elements:**

Controlling statechart

# (C7301) Controlled-termination subactivities of self-terminating activity

**Error In:**

Correctness

**Description:**

This check is generated when an activity of termination type reactive controlled (which is a subactivity of an activity of termination type reactive self) has no control activity.

**Erroneous Elements:**

Activities

# (C7303) Elements containing references to transparent activities

**Error In:**

Correctness

**Description:**

The control activity contains references to an activity defined to be a transparent level. The references should be made to the activity's nontransparent subactivities.

**Erroneous Elements:**

Activities

# Activity and Module Chart Relation Checks

This section documents the Check Model tests for activity-chart and module chart relation checks. The checks are listed in numerical order.

## (C7401) Unresolved elements

**Error In:**

Completeness

**Description:**

Activities, data-stores, and modules referenced in the properties in the fields is activity/data-store and implemented by module are not found in the expected scope in the workarea.

The modules are searched in the entire workarea; the activities and data-stores are searched only in the activity-chart that describes the implementing module.

**Erroneous Elements:**

Activities, data-stores, and modules

# (C7403) Elements that flow in the activity chart without corresponding flow in the module chart

**Error in:**

Correctness

**Description:**

Data elements are specified as flowing between activities in the activity-chart, but they do not flow between the corresponding modules in the module chart.

**Example:**

In the following figure, assume that A1 is implemented by M1 and A2 is implemented by M2. X flows between A1 and A2 in the activity-chart. X will be detected as an erroneous element because it does not flow between the modules that implement A1 and A2.
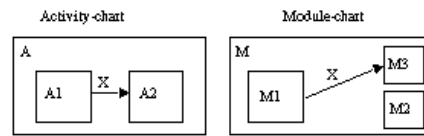


**Erroneous Elements:**

Events, conditions, data-items, and information-flows

# (C7405) Elements that flow in the module chart without corresponding flow in the activity chart

**Error In:**

Completeness

**Description:**

Data elements specified as flowing between modules in the module chart do not flow between the corresponding activities in the activity-chart.

**Erroneous Elements:**

Events, conditions, data-items, and information-flows

# (C7407) Basic activities not allocated to any module

**Error In:**

Completeness

**Description:**

There are activities that are not decomposed into other activities and data-stores, and are not specified as implemented by any module.

**Erroneous Elements:**

Activities

# (C7409) Data-stores not allocated to any module

**Error In:**

Completeness

**Description:**

There are data-stores that do not reside in any module.

**Erroneous Elements:**

Data-stores

# (C7411) Basic modules to which no activity or data-store is allocated

**Error In:**

Completeness

**Description:**

There are basic modules (descendants of the module described by the tested activity-chart) that implement no activity and in which no data-store resides.

**Erroneous Elements:**

Modules

# (C7501) Activities or data-stores whose allocation in modules violates the hierarchy rules

**Error In:**

Correctness

**Description:**

**There are activities** and data-stores whose hierarchy as specified in the activity-chart is not consistent with the hierarchy of modules to which they are allocated.

**Example:**

In the following figure, A represents an activity and M is a module. Assume that it is specified in the properties that "A is implemented by M1" and "A1 is implemented by M2". Because A1 is a subactivity of A, this does not match the relation between M1 and M2.
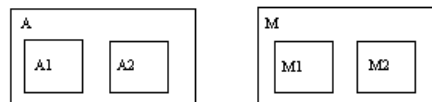


**Erroneous Elements:**

Activities and data-stores

# Index

Check Model Guide

Check Model Guide