TM Forum Technical Report

Intent Common Model - Intent Reporting

TR290B

Maturity Level: **General availability (GA)**	Team Approved Date: 04-Jul-2024
Release Status: Production	Approval Status: TM Forum Approved
Version 3.6.0	IPR Mode: RAND

Table of Contents

Notice 2
Table of Contents 4
Executive Summary 5
Introduction 6
Scope 6
Revision Information 6
1. Notation and Dependencies 7
2. The Structure of intent Reports 9
3. Reporting Scope 13
4. Referencing Intent Elements 15
5. Content of intent reports. 16
6. Basic Intent Report Content 17
6.1. Explicit Evaluation Result 17
6.2. Explanation of Evaluation Result 18
6.3. Intent Report properties 18
6.3.1. Report Timestamp 18
6.3.2. Report Sequence Number 19
6.3.3. Intent Handling and Update States 19
6.4. Expectation Report 20
6.5. Target Report 20
6.5.1. Content of Target Reports 21
6.5.2. Split of Target Reports 21
6.6. Report about Delivery Expectation 21
6.7. Report about Property Expectation 22
6.8. Report about Reporting Expectation 22
6.8.1. Event for Reporting 22
6.9. Condition Reports 23
7. Observation Intent Report Content 24
8. Additional report content specified in extension models 26
8.1. Context Reports 26
8.2. Information in Intent Reports 26
9. Administrative Appendix 27
9.1. Document History 27
9.1.1. Version History 27
9.1.2. Release History 27
9.2. Acknowledgments 28

Executive Summary

This document is part of the TR290 series specifying the intent common model as part of the TM Forum Intent Ontology. The focus of this document is the specification of how intent reports are structured and expressed.

Introduction

The TM Forum Intent Ontology (TIO) is a model of intent expression and management. A major part of the model is the intent common model specified in the TR290x series of documents. It is mandatory for an intent management function to support at least one version of the intent common model. It contains generic concepts of intent models and expression vocabulary. This means its vocabulary is not specific to an application domain or particular use case. It rather defines concepts that are broadly useful for intent expression in any domain and intent use case. The use case specific and optional models are referred to as "intent extension models" and not in scope of the TR290x series of documents.

Intent reports are used in intent based operation to keep the intent owner informed about status and progress regarding its intents. In this respect it closes the intent control loop between intent owner and intent handler. While intents specify the requirements that an intent handler needs to achieve, an intent report provides details about the actual achievement and current state of the intent handling and the underlying operated system.

Scope

This document is part of the intent common model. It specifically covers the ontology and vocabulary for expressing intent reports.

Revision Information

This revision v3.6.0 of the intent common model is part of the TM Forum Intent Ontology (TIO) v3.6.0. Changes are logged in the main document TR290.

1. Notation and Dependencies

The intent common model depends on the following models:

Model	Prefix	Namespaces	Published by	Purpose in the model
Intent Common Model	icm	`http://tio.models.tmforum.org/tio/v3.6.0/IntentCommonModel/`	TM Forum	General ontology model of intent and intent report expression. This document is part of the intent common model specification.
Intent Management Ontology	imo	`http://tio.models.tmforum.org/tio/v3.6.0/IntentManagmentOntology/`	TM Forum	Defines basic vocabulary and concepts of intent based operation. This document specifies vocabulary about intent management function and their roles, as well as the types of intent models within the TM Forum Intent Ontology (TIO).
Conditions and Logical Operators Ontology	log	`http://tio.models.tmforum.org/tio/v3.6.0/LogicalOperators/`	TM Forum	Specifies logical operators to express logical relationships and the evaluation of truth values.
Quantity Ontology	quan	`http://tio.models.tmforum.org/tio/v3.6.0/QuantityOntology/`	TM Forum	Introduces quantities and quantity operators.
Set Operators	set	`http://tio.models.tmforum.org/tio/v3.6.0/SetOperators/`	TM Forum	Specification of set operators
Function Definition Ontology	fun	`http://tio.models.tmforum.org/tio/v3.6.0/FunctionOntology/`	TM Forum	Basic expression of functions
Metrics and observation	met	`http://tio.models.tmforum.org/tio/v3.6.0/MetricsAndObservations/`	TM Forum	The metrics and observation model introduces metrics as measurable and observable items and defines vocabulary to manage observations about the metric
Time Ontology in OWL	t	`http://www.w3.org/2006/time#`	W3C	Expression of date and time [owltime]

Model	Prefix	Namespaces	Published by	Purpose in the model
RDF version 1.1	rdf	http://www.w3.org/1999/02/22-rdf-syntax-ns#	W3C	Providing fundamental modeling artifacts [rdf11]
RDF Schema 1.1	rdfs	http://www.w3.org/2000/01/rdf-schema#	W3C	Providing fundamental modeling artifacts [rdfs11]
XML Schema	xsd	http://www.w3.org/2001/XMLSchema#	W3C	Providing of data types for literal objects [xsd-1] [xsd-2]
Examples	ex	http://www..example.org/	IANA	Reserved domain name for examples

Table 1: Model references

The intent common model is based on the Resource Description Framework (RDF) [rdf, rdf_mt, rdf_primer] and the Resource Description Framework Schema (RDFS) [rdfs] published by the World Wide Web Consortium (W3C).

Furthermore, the intent common model depends on base models of the TM Forum intent ontology. These are the intent management ontology, the function definition ontology, the logical operators ontology and the quantity ontology.

2. The Structure of intent Reports

Intent reports are instances of the class `icm:IntentReport`.

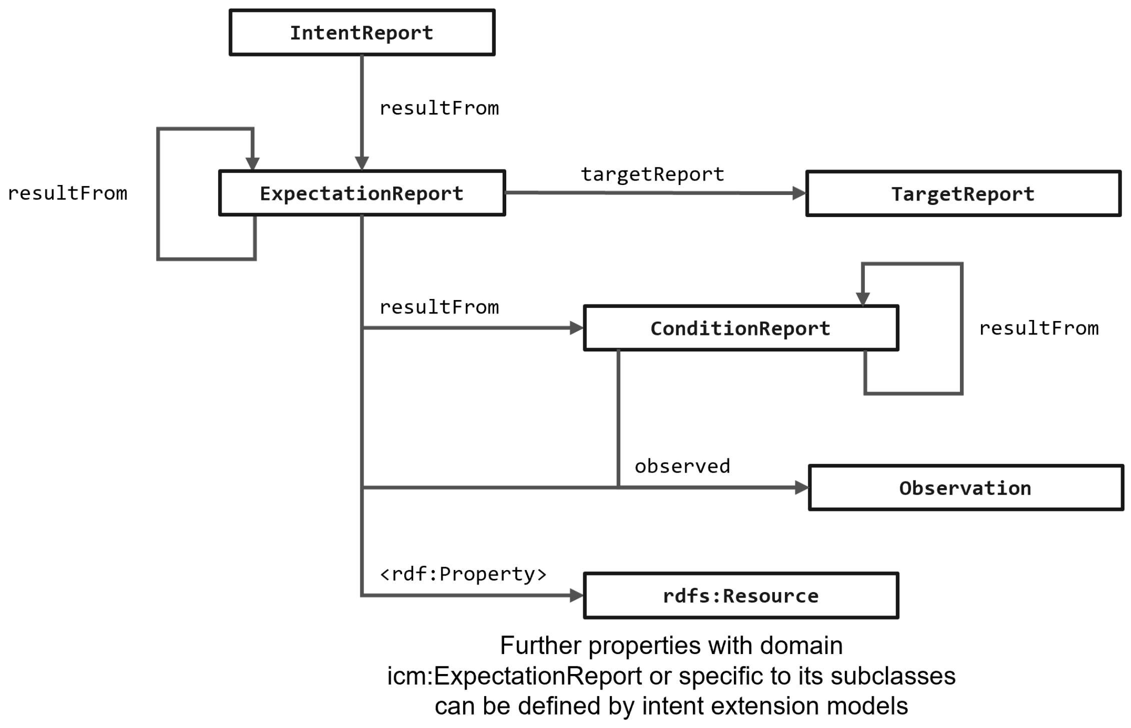

Figure 1: Structure of Intent Reports

The property `icm:resultFrom` associates an intent report with an expectation report that contains details about why the compliance evaluation result of the intent is "true" indicating compliance or false indicating degradation. Furthermore, the property `icm:resultFrom` can be used to express statements with instances of expectation reports or condition reports as subject and object. This means the property `icm:resultFrom` can be used to reflect the hierarchy of expectations and conditions in the intent with expectation and expectation reports in the intent report.

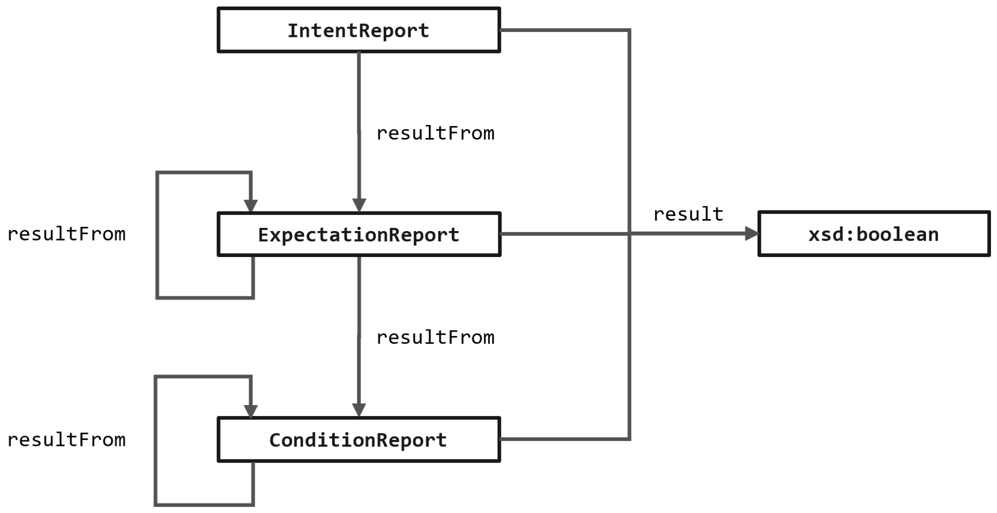

Figure 2: Providing evaluation results

Intent reports might not contain detailed reports about all elements of the intent depending on the reporting scope set by a reporting expectation. Contributors to the compliance result stated with the property icm:resultFrom would only include intent element reports that are provided within this intent report. This might therefore be incomplete and miss out contributing factors to the result.

The property `icm:result` states the evaluation result of the intent element at the time of report generation. It is a boolean truth value. For intent and expectation objects `icm:result` is interpreted as compliance to the requirements or subsets of requirements. For conditions, it simply states the truth value. In general the `icm:result` property can be used for report types for every intent element that has an associated truth value. In the intent common model those are intents, expectations and conditions. The property `icm:result` is therefore used with intent reports, expectation reports and condition reports.

For example:

```
ex:Intent1
  a icm:Intent ;
  log:allOf ( ex:E1 ex:E2 )
.
ex:E1
  a icm:PropertyExpectation ;
  icm:anyOf ( ex:C1 ex:C2 )
.
ex:E2
  a icm:PropertyExpectation ;
  icm:anyOf ( ex:C1 ex:C3 )
.
ex:C1
  a log:Condition ;
.
ex:C2
  a log:Condition ;
  icm:allOf ( ex:C1 ex:C3 )
```

```
.
ex:C3
  a log:Condition ;
.
```

This is an example intent with two expectation and three condition instances. In this example only the structure and dependencies of the intent element is shown, and the detailed requirements are left out. A respective complete intent report for this intent would be:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  icm:result true ;
  log:resultFrom ex:ER1, ex:ER2
.
ex:ER1
  a icm:ExpectationReport ;
  icm:about ex:E1 ;
  icm:result true ;
  icm:resultFrom ex:CR1, ex:CR2
.
ex:ER2
  a icm:ExpectationReport ;
  icm:about ex:E2 ;
  icm:result true ;
  icm:resultFrom ex:CR1, ex:CR3
.
ex:CR1
  a icm:ConditionReport ;
  icm:about ex:C1 ;
  icm:result true ;
.
ex:CR2
  a icm:ConditionReport ;
  icm:about ex:C2 ;
  icm:result false ;
  icm:resultFrom ex:CR1, ex:CR3
.
ex:CR3
  a icm:ConditionReport ;
  icm:about ex:C3 ;
  icm:result false ;
.
```

This intent report contains two expectation reports and tree condition reports about the two expectations and three conditions in the intent. The property icm:about refers to the respective element in the intent the report element is associated with and reports about.

The property `icm:result` shows the evaluation result of each element at the point in time when the intent got generated. It is a boolean truth associated with the intent element. For intent and expectations, this truth value is interpreted as compliance.

The property icm:resultFrom refers to the element in the intent report that contributed to the evaluation result. In this example both expectations did contribute to the result of the intent. Thus, the intent report states that the result is from the two expectation reports. Respectively, the expectation reports state the condition instances they gain their evaluation result from. And the condition `ex:C2` evaluates `ex:C1` and `ex:C3` to obtain its result. This means that the respective condition report `ex:CR2` states that the result for `ex:C2` is from the conditions `ex:C1` and `ex:C3` represented by the condition reports `ex:CR1` and `ex:CR3`.

The property `icm:targetReport` is used to assign a target report to an expectation report. Targets in an intent can contain multiple resources and furthermore, a dynamic selection of resources to be used as targets is possible. This means that intent reports my need to specify, which resources were considered to be part of a target at the moment of intent report generation. Furthermore, the intent targets might need to be separated, if its constituent resources require different reporting. This can for example be necessary if some resources are compliant to the requirements, but others are not. Chapter 8 discusses expectation reports including target reports in detail.

The property met:`observed` is used to assign observations to an element of an intent report. For example, it allows stating the value observed for a KPI at the time of report generation. This is further explained in Chapter 10.

Further properties of expectation reports can be introduced by intent extension models to report about respective additional properties they introduce within an intent.

3. Reporting Scope

The reporting expectation in the intent defines the reporting scope through its target. As a general rule, the intent report shall only contain reports for intent elements that are members of the reporting expectation's target container. This means for example that expectation reports are only provided in the intent report for expectations that were members of the target in the respective reporting expectation. The same rule holds for pairs of conditions and condition reports, targets and target reports as well as context and context reports. The only exception is intent. An intent report instance is always the base and provided regardless if the intent was part of the target or not.

For example:

```
ex:Intent1
  a icm:Intent ;
  log:AnyOf ( ex:E1 ex:E2 ex:E3 )
.
ex:E1
  a icm:Expectation ;
.
ex:E2
  a icm:Expectation ;
.
ex:T1
  a icm:Target ;
  rdfs:member ex:Intent1, ex:E1
.
ex:E3
  a icm:ReportingExpectation ;
  icm:target ex:T1
.
```

This example intent contains three expectations. One of them is the reporting expectation. The target of the reporting expectation has only two members, the intent ex:Intent1 itself and the expectation ex:E1. The resulting intent report would be:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  icm:result true
  icm:resultFrom ex:E1 ;
.
ex:ER1
  a icm:ExpectationReport ;
  icm:about ex:E1 ;
  icm:result false
.
```

The intent report contains only explicit reports about the intent elements that were explicitly mentioned in the target of the reporting expectation. In this example the evaluation result for the intent is "true", while the only reported expectation is "false". The reporting expectation did not ask for a report about the expectation ex:E2, although its result has an impact on the overall compliance evaluation.

The scoping of intent reports is quite flexible, but the example above shows that it needs to be used sensibly to avoid reporting gaps.

It is only possible to report about elements of the intent that are represented with an IRI/URI. This means, it is not possible to report explicitly about blank nodes in the intent graph, even if they represent instances of classes that are usually reported about. For example, a condition introduced as a blank node cannot get an individual condition report within the intent report. This means practically that all major intent elements that are of interest in intent reporting should not be expressed using blank nodes.

4. Referencing Intent Elements

Intent reports provide status and progress update about intents and their elements. Each element in an intent can be matched with a respective element in the intent report. For example an individual expectation would be considered with an individual expectation report.

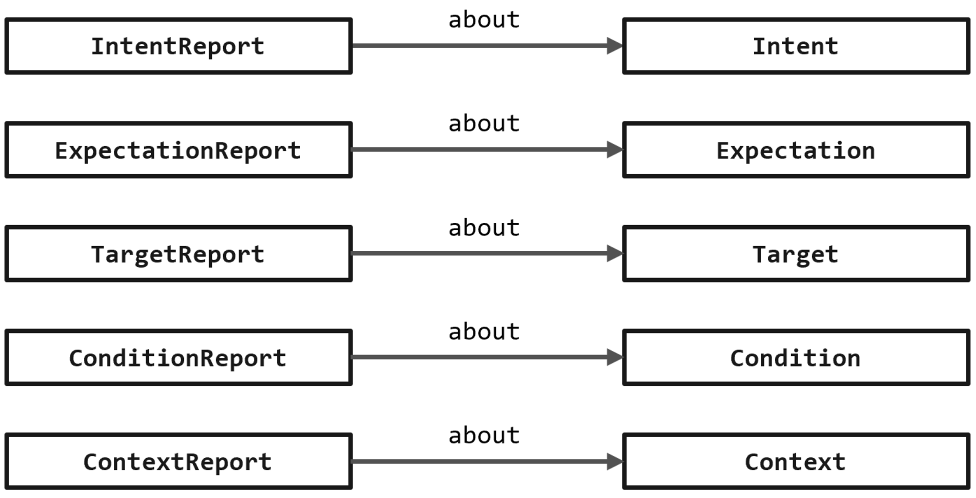

Figure 3: Referencing to the reported intent element

The property `icm:about` references the intent element the intent report element is reporting about. Its range contains typically the intent element individual equivalent to the intent report element individual. For example, an intent report would use `icm:about` to refer to the intent. For example:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1
.
ex:CR1
  a icm:ConditionReport ;
  icm:about ex:C1
.
```

This example shows the use of icm:about. It expresses the intent report ex:IR1 provides a report about the intent ex:Intent1. It is used to link the condition report ex:CR1 to the condition ex:C1 it is reporting about.

5. Content of intent reports.

The content of an intent report refers to the information provided within it for all intent elements that are in scope. This includes basic information such as the timestamp of report generation or the serial number of an intent report. It can then include further operational information regarding the intent such as intent handling state or information regarding compliance to the intent requirements.

The content provided in the intent report depends on the class or classes of the reporting expectation that caused the intent report generation as explained in part A of the Intent Common Model [TR290A]. Basic content is included for intent reports generated for expectations of class `icm:ReportingExpectation` and all of its subclasses. This also means that the content provided in intent reports generated for `icm:ObservationReportingExpectation` contains at least the content of basic reporting, because `icm:ObservationReportingExpectation` is a subclass of `icm:ReportingExpectation`. However, content provided for `icm:ObservationReportingExpectation` can include additional information such as the latest observation for metrics used within the intent.

The following chapters describe possible intent report content for reporting expectation classes introduced by the intent common model. This includes the parent class `icm:ReportingExpectation` leading to basic content and the subclass `icm:ObservationReportingExpectation` with its additional content about observations for metrics used.

Further subclasses of `icm:ReportingExpectation` might be added by intent extension models, which then would then also define further content specific to the extension.

6. Basic Intent Report Content

This chapter describes the content of intent reports generated for reporting expectations of class `icm:ReportingExpectation` and its subclasses.

6.1. Explicit Evaluation Result

Some elements within the intent have an associated boolean value. It can come from the evaluation of conditions or aggregation of results through logical operators. Examples are instances of `log:Condition`, where the boolean value states the evaluation result of the condition expression. Also, intent and expectation instances have an associated boolean value, which is also interpreted as compliance of the operated system and its resources to the requirements they define.

The property `icm:result` states the boolean value associated with the individual it is reporting about at the time of report generation. Its range is a literal with datatype `xsd:boolean`.

The interpretation of `icm:result` depends on its subject. If used within a condition report, it would state the overall evaluation result of the respective condition in the intent.

It can also be used as property of intent reports and expectation reports. The truth value of their associated intent and expectation individuals expresses compliance of the managed system and its resources to the requirements defined in intent or partial requirements defined by expectations. Therefore, the boolean value "`true`" stated by `icm:result` as property of intent reports or expectation reports indicates compliance to the requirements. Respectively, the value "`false`" indicates non-compliance.

For example:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  icm:result true
.
ex:ER1
  a icm:PropertyExpectationReport ;
  icm:about ex:E1 ;
  icm:result true
.
ex:CR1
  a icm:ConditionReport ;
  icm:about ex:C1 ;
  icm:result false
.
```

This example defines that intent report `ex:IR1` is reporting about the intent ex:Intent1. It further defines the expectation report `ex:ER1` reporting about the expectation `ex:E1`. Both were evaluated as "true" at the moment of report generation. This means that the operated system and its resources did meet all requirements. This example also shows the condition report `ex:CR1` for condition `ex:C1` from the intent. It was evaluated as false, indicating that the condition was not met when the intent report was generated.

As it is a condition, it does not immediately imply compliance or any other interpretation of the result.

In TURTLE the strings "`true`" and "`false`" are predefined literal values of type `xsd:boolean`.

6.2. Explanation of Evaluation Result

The intent handler can include explanations of why the evaluation result is true or false and therefore why a requirement is fulfilled or not.

A reason is represented by individuals of class `icm:Reason` or its subclasses. The definition of an individual reason would use the `rdf:label` and `rdf:comment` properties to provide human-readable descriptions of what it represents and implies. Further properties might be used to model additional explanations and information. They might be defined by intent extension models.

The property `icm:reason` would assign a reason to an intent report element. In general a reason can always make sense to provide in addition to an `icm:result` property as it raises the understanding of why the evaluation has produced the result value.

A catalog of pre-defined reasons will be added in future updates of the TM Forum Intent Ontology. Additional reasons might be defined by intent extension models.

6.3. Intent Report properties

The class `icm:IntentReport` can have the following properties providing general information about an individual report and the intent handler:

6.3.1. Report Timestamp

The property `icm:reportGenerated` assigns the timestamp of report generation to an intent report instance. The object of this property is of class `t:Instant`. It is an instant defined by the time ontology in OWL [owltime] and refers to a point in time.

The timestamp of an intent report reflects the date and time of report generation and therefore all values and results stated in the intent report are the observed state or measurement known by the intent handler in its knowledge base at that point in time.

The timestamp can be expressed using properties specified in the time ontology in OWL or by using time related data types of XML Schema.

For example:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  icm:reportGenerated [ t:inXSDDateTimeStamp "2023-02-
06T13:30:10+01:00"^^xsd:dateTime ]
.
```

This example assigns a time stamp to the intent report. It expresses that the intent report was generated on the 6th of February 2013 at 13:30 within the timezone UTC plus one hour. This example has used the `xsd:dateTime` datatype of XML Schema. The time ontology in OWL allows other formats for specifying a time instant, and they can also be used in intent and intent reports.

6.3.2. Report Sequence Number

The property `icm:reportNumber` assigns a sequence number to the intent report instance. It starts with 1 for the first report generated for the intent, and it is increased with every subsequent report that is sent for this intent. The range of the property `icm:reportNumber` is a positive integer.

In case of complex intent reporting setups with multiple report receivers and multiple reporting expectations, it might happen that not every intent owner receives all intent reports that were generated for the intent. This means an intent owner cannot assume there is an issue, if there are gaps in the sequence of the reports it has received. It can however conclude that higher numbers indicate reports that were generated after those with lower numbers.

For example:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  icm:reportNumber "1"^^xsd:positiveInteger
.
```

This example states that `ex:IR1` is the intent report for intent `ex:Intent1` with sequence number 1. Therefore, it is the first intent report generated for this intent.

6.3.3. Intent Handling and Update States

The intent handler maintains handling and update state machines for every intent as described in the specification of Intent Management State Machines [TR292B]. An intent report provides the state of the intent handler with respect to each of these state machines at the time of intent report generation. This is expressed in intent reports using the properties `imo:handlingState` and `imo:updateState`.

The intent handling and update states also imply decisions of the intent handler regarding acceptance or rejection of the intent or its updates.

If a respective state machine is not started at the time of reporting, its state is not included in the intent report.

For example:

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  imo:handlingState imo:StateCompliant ;
  imo:updateState icm:StateNoUpdate
.
```

This example intent report states that the handling state for the intent `ex:Intent1` is `imo:StateCompliant` and the update state is `imo:StateNoUpdate`. This means that the managed system is overall compliant to the intent and no intent update process is currently ongoing.

6.4. Expectation Report

The class `icm:ExpectationReport` with its subclasses represents the reports for expectations defined in an intent.

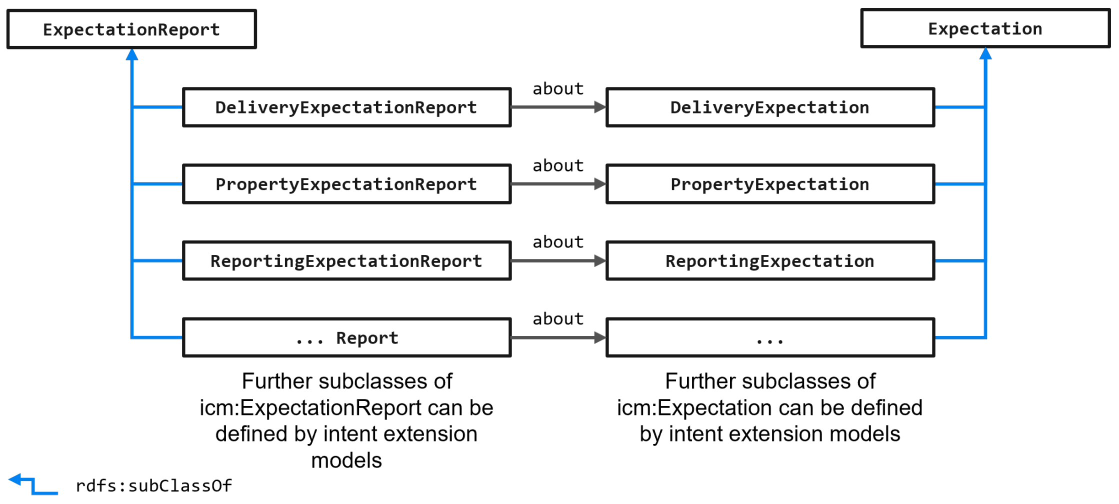
ExpectationReport
Expectation
DeliveryExpectationReport
about
DeliveryExpectation
PropertyExpectationReport
about
PropertyExpectation
ReportingExpectationReport
about
ReportingExpectation
... Report
about
...
Further subclasses of icm:ExpectationReport can be defined by intent extension models
Further subclasses of icm:Expectation can be defined by intent extension models
rdfs:subClassOf

Figure 4: Expectation report classes and their relationship to expectations

The reporting for expectations depends on the expectation targets. An expectation report always has at least one target assigned to it. It refers to the concrete instance that is used for fulfillment of the intent. The general rule is that a separate expectation report is generated for each member resource of the target container. If multiple target member resources would have the same report, they can be reported collectively.

The evaluation of compliance is typically based on observations about the target to express and therefore evaluate compliance. A target resource that is not available would mean no observations can be made and compliance cannot be evaluated. This is interpreted as not compliant. This means an unknown target resource would indirectly lead to reporting of a degradation.

6.5. Target Report

The report for a target is provided by an instance of class `icm:TargetReport`. It is assigned to an Expectation Report using the property `icm:targetReport`.

A target in an intent is a container of resources supposed to be used to fulfill the requirements specified by expectations. The target might be defined dynamically with resources changing over time and the intent handler might have made a selection of resources to be used. Therefore, reporting about a target primarily means enumerating the target resources that were used to fulfill the requirements at the time of report generation. A target report is therefore defined as a container of all resources that were selected and used to fulfill requirements.

6.5.1. Content of Target Reports

A target report explicitly enumerates its members. It does not refer to members through abstract functions. This behavior is important, because the results of an abstract function are dynamically changing over time. Consequently, an abstract function would potentially refer to different sets of members at different points in time. A target report has to provide and preserve the information what the target resources are at the time of reporting.

On the other hand it is also not a good solution to individually enumerate all members that were assigned to a target through an abstract function, because the number of target resources might be very large and therefore requires a lot of processing and memory to create. For example, a target might specify that all RBS sites are targets and there can be thousands of them in a network. Enumerating them in every intent report would make the intent report very large. For this reason a target report only enumerates the target resources that were already enumerated explicitly in the target specification. In addition, the target report would provide the number of member resources the target did contain at the time of report generation. The property `icm:targetCount` provides this information.

6.5.2. Split of Target Reports

If all member resources of the target in the intent fulfill an expectation in the same way, there is only one target report instance and only one related expectation report instance needed. However, subsets of target resources might have different results. If some resources from the target fulfill the requirement and others do not, the reporting is split into multiple distinct expectation reports. They are connected to target reports representing a subset of target resources specified in the intent. This also means that multiple target reports might be created and connected to multiple distinct variations of expectation reports to report about one target and one expectation in the intent. One expectation report might state compliance for a subset of target resources, while another expectation report for the same expectation would state non-compliance for another subset of target resources. In many cases it is actually needed to create an individual partial target report with only one member. This is needed if the expectation states measured KPIs and the values measured for these KPIs are different for different target resources.

6.6. Report about Delivery Expectation

The report for a delivery expectation is provided by an instance of class `icm:DeliveryExpectationReport`.

A delivery expectation according to the intent common model uses the properties `icm:deliveryType` and `icm:deliveryDescription` to describe the requirement of what shall be delivered. The report for a delivery expectation can only confirm whether these requirements are met or not by whatever target resources were chosen. This means that the property `icm:result` is the main information provided by delivery expectation reports. However, a delivery expectation usually involves dynamic selection of target resources. Therefore, a target report stating the used and selected resources is typically essential in combination with a delivery expectation report.

If the delivery expectation is not fulfilled, the reason provided through the property `icm:reason` might also be interesting. It can for example state that provisioning of a new service instance is still pending or that it has failed. The first would be an indication that the situation is most likely temporary and should be solved soon. The latter might indicate a more serious issue that needs attention.

Intent extension models can define further properties to be used with a delivery expectation. These models must also specify the respective reporting for the new properties in a delivery expectation report.

6.7. Report about Property Expectation

The report for a property expectation is provided by an instance of class `icm:PropertyExpectationReport`.

The requirements of a property expectation are typically provided using condition objects and logical interconnection of conditions. The property icm:result states the overall evaluation of this condition based logic for an instance of a property expectation at the moment of intent report generation.

Reports for distinct contributing conditions can be provided with condition reports. The property expectation report can refer to these contributions using the `icm:resultFrom` property.

Intent extension models can define further properties to be used with a property expectation. These models must then also specify the respective reporting for the new properties in a property expectation report.

6.8. Report about Reporting Expectation

The report for a reporting expectation is provided by an instance of class `icm:ReportingExpectationReport`.

The target report for the target of a reporting expectation contains the set of intent report elements that match each intent element in scope.

Intent extension models can define further properties to be used with a reporting expectation. These models must also specify additional properties that may be needed in the respective report in a reporting expectation report.

6.8.1. Event for Reporting

An intent report is initiated by events in the intent handler. The reporting expectation report states which event it originated from using the property imo:event. defined in "Intent Management State Machines" [TR292B]. For example.

```
ex:IR1
  a icm:IntentReport ;
  imo:event ex:Degrades123
.
```

This example shows an intent report that states which event has caused its generation. The individual event `ex:Degrades123` is an instance of class `imo:Degrades`. As such it expresses that the intent handling state machine has assumed the state `imo:StateDegraded`. This is also an indication that not all requirements specified by the intent are fulfilled.

If for some reason multiple events have occurred simultaneously or in short succession, the intent handler can choose to only generate a single intent report. In this case multiple `imo:event` statements would be added to the intent report.

6.9. Condition Reports

The report for a condition object is provided by an instance of class `icm:ConditionReport`.

The main property of a condition report is `icm:result`. It provides the overall evaluation result of all contributing conditions associated with the condition object reported about. The property `icm:resultFrom` would then refer to the condition reports of contributing conditions.

Furthermore, condition reports can state observations the evaluation is based on.

7. Observation Intent Report Content

If an intent report is created for a reporting expectation of class `icm:ObservationReportingExpectation`, the intent shall contain details about the metrics used to evaluate the intent in addition to basic intent report content.

Observations are individuals of class `met:Observation` defined in the metrics and observation ontology [TR292G]. In intent reports observation objects are used to represent a measured or otherwise observed fact about the targeted resource at the moment of intent report generation. Examples are KPI values, aggregated insights or resource states. The intent common model uses further vocabulary from the metrics and observations model:

The property met:observed is used to associate an observation object with the intent element report it is contributing to. This can for example be a condition report or an expectation report. The observation should be provided with the report that represents the intent element that has directly used the observed metric, KPI, state, etc.

The property met:observedMetric states what was observed. This referees, for example, to the metric, KPI or state for which a value is provided. The observed value is stated using the generic `rdf:value` property.

Intent extension models can define further properties to be used with an observation object to cover the different needs of metrics or KPIs.

For example:

```
ex:Intent1
  a icm:Intent ;
  log:allOf ( ex:E1 ex:E2 )
.
ex:E1
  a icm:PropertyExpectation ;
  icm:target ex:T1 ;
  log:allOf ( ex:C1 )
.
ex:C1
  a log:Condition ;
  quan:smaller ( ex:Latency1
               "10ms"^^quan:quantity
             )
.
```

This is the intent used in the example. It specifies a condition regarding latency contributing to a property expectation. The example is simplified and leaves out elements not in focus of this chapter, such as the target definition or additional expectations.

```
ex:IR1
  a icm:IntentReport ;
  icm:about ex:Intent1 ;
  icm:result true ;
  icm:resultFrom ex:ER1
.
ex:ER1
```

```
  a icm:PropertyExpectationReport ;
  icm:about ex:E1 ;
  icm:result true ;
  icm:resultFrom ex:CR1
.
ex:CR1
  a icm:ConditionReport ;
  icm:about ex:C1 ;
  icm:result true ;
  met:observed [ a met:Observation ;
               met:observedMetric ex:latency ;
               rdf:value "8ms"^^quan:quantity
             ]
.
```

This is a possible report for the intent provided above. It states that the intent and the expectation are compliant by providing the result of the compliance evaluation as boolean values. Condition report states the evaluation result of the condition. It also provides the observed value that did lead to the condition result by using an observation object. It states that `ex:Latency1` was the observed topic. This is the metric used in the condition of the intent. The observation object furthermore states the value observed. It is the latest value known at the time of report generation.

8. Additional report content specified in extension models

The intent common model has introduced generic classes for context and information without providing use case specific details. This gap would be filled by intent extension models. The intent common model also defines generically how context and information can be represented and used in intent reports. However, details for the reporting shall also be specified by the respective intent extension models. An significant part of this specification is the definition of further subclasses of `icm:reportingExpectation`. These subclasses would imply additional content to be added to intent reports.

8.1. Context Reports

A report about a context is provided by an instance of class `icm:ContextReport`. The property `icm:contextReport` can be used to assign context reports to elements of the intent report that reports about the elements of an intent that had the respective context.

Specific types of contexts are defined by intent extension models introducing subclasses and sub properties of icm:Context and icm:context. These intent extension models would then also define respective subclasses and sub properties of `icm:ContextReport` and `icm:contextReport`. In addition, it would specify what information should be provided in a context report and how it can be expressed.

8.2. Information in Intent Reports

Information can be assigned to any element of the intent report. The class icm:Information and its subclasses as well as the property icm:information and its sub properties as defined in the intent common model and further detailed by intent extension models can be used to attach additional information to elements of the intent report.

9. Administrative Appendix

9.1. Document History

9.1.1. Version History

Version Number	Date Modified	Modified by:	Description of changes
3.0.0	07-Feb-2023	Alan Pope	Final edits prior to publication
3.1.0	11-Apr-2023	Alan Pope	Final edits prior to publication
3.2.0	11-Apr-2023	Alan Pope	Final edits prior to publication
3.4.0	29-Feb-2024	Alan Pope	Final edits prior to publication
3.5.0	03-May-2024	Alan Pope	Final edits prior to publication
3.6.0	04-Jul-2024	Alan Pope	Final edits prior to publication

9.1.2. Release History

Release Status	Date Modified	Modified by:	Description of changes
Pre-production	07-Feb-2023	Alan Pope	Updated to version 3.0.0
Pre-production	17-Mar-2023	Adrienne Walcott	Updated to Member Evaluated status
Pre-production	11-Apr-2023	Alan Pope	Updated to version 3.1.0
Pre-production	15-May-2023	Adrienne Walcott	Updated to member Evaluated status
Pre-production	15Aug-2023	Alan Pope	Updated to version 3.2.0
Pre-production	18-Sep-2023	Adrienne Walcott	Updated to member Evaluated status
Pre-production	29-Feb-2024	Alan Pope	Updated to version 3.4.0
Production	26-Apr-2024	Adrienne Walcott	Updated to reflect TM Forum Approved status
Pre-production	03-May-2024	Alan Pope	Updated to version 3.5.0
Production	28-Jun-2024	Adrienne Walcott	Updated to reflect TM Forum Approved status
Pre-production	04-Jul-2024	Alan Pope	Updated to version 3.6.0
Production	30-Aug-2024	Adrienne Walcott	Updated to reflect TM Forum Approved status

9.2. Acknowledgments

Team Member (@mention)	Company	Role*
Jörg Niemöller	Ericsson	Author, Project Co-Chair
Kevin McDonnell	Huawei	Project Co-Chair
Yuval Stein	Amdocs	Project Co-Chair
Kamal Maghsoudlou	Ericsson	Key Contributor
Leonid Mokrushin	Ericsson	Key Contributor
Marin Orlić	Ercisson	Key Contributor
Aaron Boasman-Patel	TM Forum	Additional Input
Alan Pope	TM Forum	Additional Input
Dave Milham	TM Forum	Additional Input
Xiao Hongmei	Inspur	Reviewer

**Select from: Project Chair, Project Co-Chair, Author, Editor, Key Contributor, Additional Input, Reviewer*