TM Forum Exploratory Report

Using Intent in AN - Use Cases, Scenarios and Examples

IG1253E

Maturity Level: **Alpha**	Team Approved Date: 08-Aug-2023
Release Status: Pre-production	Approval Status: Member Evaluated
Version 1.0.0	IPR Mode: RAND

Notice

Direct inquiries to the TM Forum office:

181 New Road, Suite 304
Parsippany, NJ 07054, USA
Tel No. +1 862 227 1648
TM Forum Web Page: www.tmforum.org

Table of Contents

Executive Summary

Intents comprise the requirements, goals, and constraints in a simplified manner that is abstract from complex technical definitions and dependencies. To put it even simpler, intents are the "what", not the "how". The "how" is left to the "handler", which we call the "Intent Management Function". We express these requirements with a common grammar or vocabulary that we call “TM Forum Intent Ontology”, or TIO for short. Autonomous systems can only adapt to requirements if they know them. So it's very important that we can express requirements in a formal (machine-readable) and unambiguous vocabulary, i.e., an ontology.

1. A Simple Example of using Turtle

Turtle is a syntax for expressing information in the Resource Description Framework (RDF) format, which is a standard for representing data on the web. Here is a simple example using Turtle syntax:

```
@prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix foaf: <http://xmlns.com/foaf/0.1/> .

<http://example.org/people#Alice>
    a foaf:Person ;
    foaf:name "Alice" ;
    foaf:mbox <mailto:alice@example.org> ;
    foaf:knows <http://example.org/people#Bob> .

<http://example.org/people#Bob>
    a foaf:Person ;
    foaf:name "Bob" ;
    foaf:mbox <mailto:bob@example.org> .
```

Code Block 1 A Simple Example Using Turtle

This example defines two people, Alice and Bob, and some information about them. The first three lines define some prefixes that are used to abbreviate certain URI references. The rest of the example defines two resource descriptions, one for Alice and one for Bob. Each description consists of a set of triples that provide information about the person.

In this example, the triples use the following predicates:

- `a` is a shorthand for the `rdf:type` predicate, which is used to specify the type of a resource. In this case, both Alice and Bob are identified as `foaf:Persons`.
- `foaf:name` is a predicate that specifies the name of a person.
- `foaf:mbox` is a predicate that specifies the email address of a person.
- `foaf:knows` is a predicate that specifies a person that the subject knows.

2. Project Scope and Objectives

2.1. Scope

This guide serves as an interim guide to provide a user manual for the use of the TM Forum Intent Ontology (TIO). A more complete (and updated to v3 of TIO) of the user manual is expected in the future.

3. Intent-Driven Autonomous Networks (IDAN)

In the Intent-driven Autonomous Networks Phase 2 catalyst the catalyst project team (8 CSPs, 5 vendors, 1 research institute, and 1 university) tested the development of intent interfaces and models in an end-to-end use case that supports dynamic pricing for connectivity services, based on the availability of network resources and the maximum price a customer will pay. They deployed the intent interface (the candidate TMF921 Intent API) and the associated TM Forum Intent Ontology in all three operational layers - the business, service, and resource layers.

3.1. Dynamic Pricing for Connectivity Service Use Case

The catalyst use case demonstrates a dynamic pricing scenario that provides zero-touch customer experience and self-healing capabilities leveraging intent. This scenario applies to customers who require non-real-time connectivity, such as large research institutions or hospitals, who need to periodically synchronize a large amount of data with a data center or cloud. In this mode, customers can obtain higher-experience services with discounted prices. Furthermore, service providers can better utilize idle bandwidth resources to obtain more value without disrupting existing business.

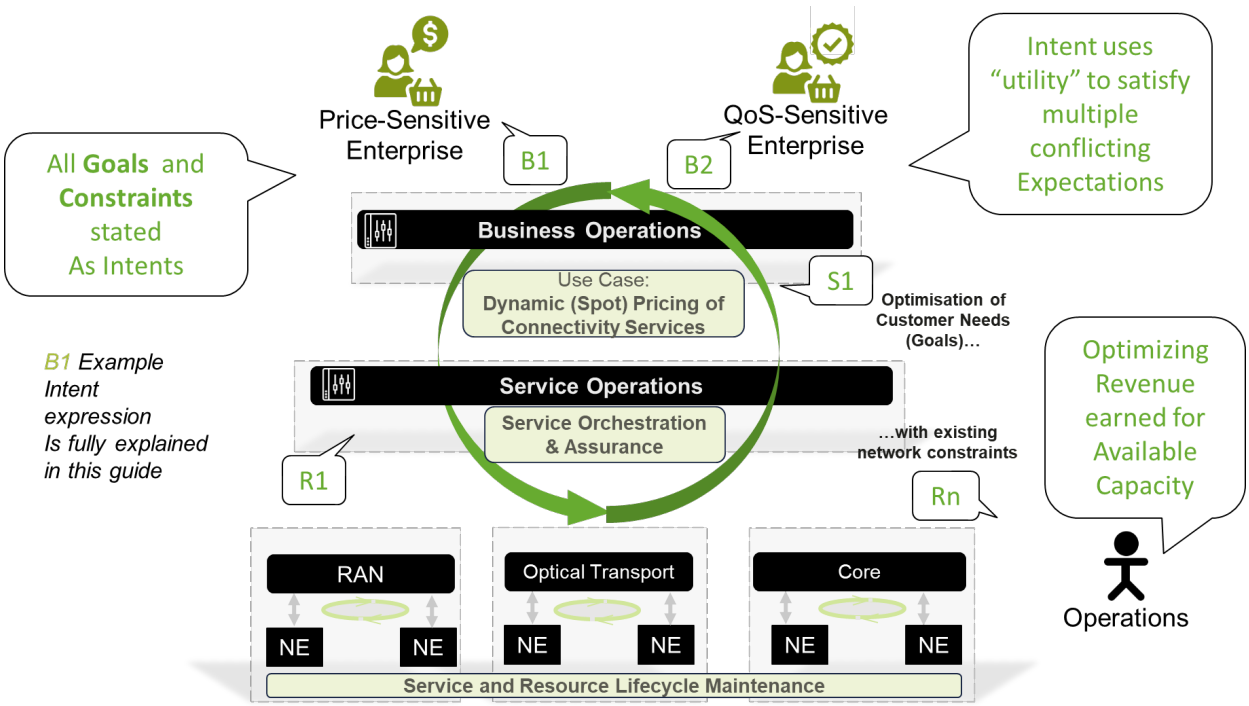

Figure 1: IDAN Phase 2 use case overview

- **Business layer** – refers to the "spot pricing" business model to enable users to obtain services with a higher user experience at lower tariffs.
- **Service layer** – translates the service intent through the Intent Management Function (IMF), creates the service order for cross-domain and cross-vendor service orchestration and works with service-based assurance to close the assurance loop.
- **Resource layer** – uses intent-based interfaces to drive fast service provisioning and network slicing at the network layer, and triggers agile and intelligent fault diagnosis and rectification based on alarms.

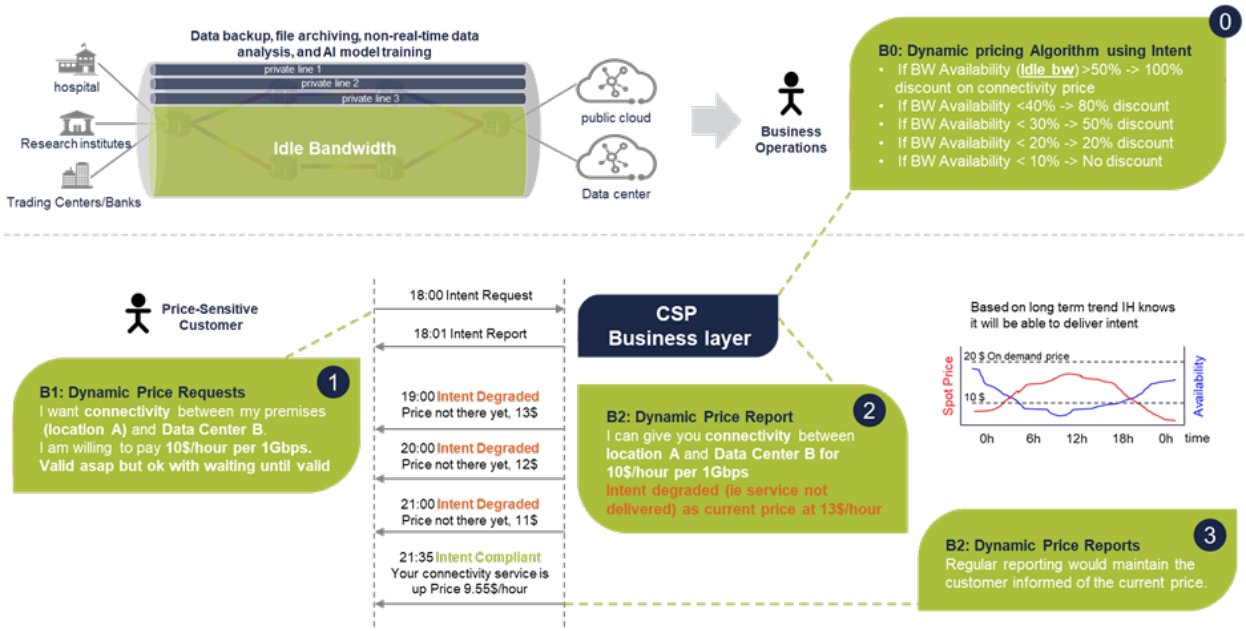

Figure 2: Dynamic Pricing Scenario

In our catalyst scenario the service provider is offering discounted prices for connectivity services based on network utilization. As shown in the following figure:

- Higher level of discount is provided based on the amount of free resources (idle bandwidth) in the network as indicated by utilization metrics. This pricing algorithm is configured in the system by the business operators using a constraint (or system wide) intent.
- Step 1: A price-sensitive customer will request a connectivity service. The customer will set the maximum price that the enterprise is willing to pay and will provide an intent request (we call this intent the B1 intent). This intent request will also express the fact that the customer is happy to wait for the service, therefore the intent should not be rejected if the resources are not available yet or the price constraint cannot be met.
- The service provider will receive the intent request and accept it. It will provide regular updates to the intent owner (the price sensitive customer) through intent reports informing the enterprise of the current price for connectivity and whether the intent is compliant or degraded. It is visible from the reports if degradation is caused by not delivering the service, because the price is too high, or for other reasons.

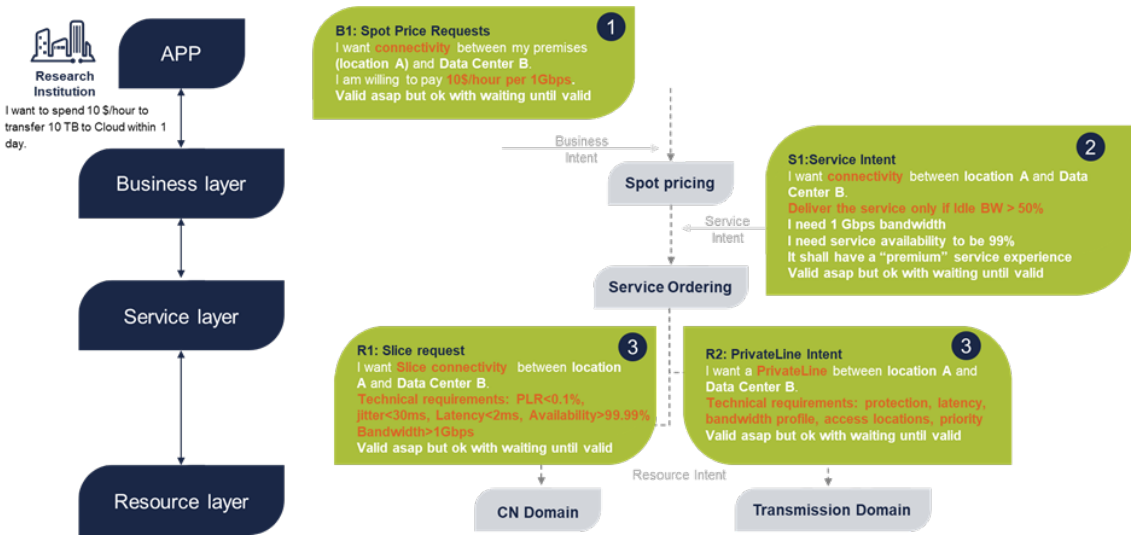

Figure 3: Spot price capability request using intent and subsequent intents in the network.

Intent is "changing" from the business to the service and eventually the resource layer, and it becomes more technical the closer it is to the network. However, for all cases we use the same API as the expressiveness of the TIO (TMF Intent Ontology) allows us to unambiguously define all these intents so that they can be fully understood by all the IMFs that will be handling the intents.

The business layer intent manager is creating the requirements it needs from the resource layer. It contains the quality of service goals, and it also contains a condition for delivering the service. While the customer has told the business intent layer about a price constraint, the service intent contains a network utilization based constraint. The pricing policy on business intent layer translates a price-based requirement into a suitable utilization-based requirement. This is necessary for a clean separation of concerns. Only the business layer has the knowledge and logic to deal correctly with pricing and other financial concerns. The service layer requirements shall be expressed using metrics inherent to service handling, which excludes any pricing.

The service layer receives the service intent (S1) and immediately accepts it, even if the network utilization criterion is not met and thus the service cannot be delivered. This means the system state with respect to this intent is degraded. Intent reports are used to communicate the intent handling state and intent acceptance back to the business layer.

In the cognitive loop of the service intent manager the network utilization is monitored, and eventually it becomes low enough for delivering the service. The policy logic in the service intent manager selects and configures a suitable service from the service catalog. In the catalyst use cases we have a connectivity service that can have three quality of service levels: Gold, Silver and Bronze. The proposal and evaluation policies have knowledge, what QoE results can be reached with each of the levels and at which resource usage. The intent manager will select silver level, because bronze level is expected to not match the required bandwidth, availability and latency requirements, while gold level is expected to bind more resources. Silver level is just right for the given requirements.

After this service is selected and configured, it is ordered by the intent management function from the service orchestrator. The orchestrator logic for this service is executing all needed actions. In this demo use case it involves the creation of the resource intent. We therefore have service intent leading to the selection of an appropriate service, which leads to the creation of the subsequent intent with requirements for the next system layer.

On the resource layer a very similar logic is analyzing the resource intents and selects an appropriate resource service. In our demo a slice resource is used. The actions of the resource orchestrator do not create further intent, but dispatch all actions needed for slice setup and configuration.

3.2. Intent Degradation and Healing Actions

In our use case demo we show the system's reaction to an issue found.

Based on observations in data and assurance rules the issue is detected in service and resource assurance and reported to the knowledge base. The intent manager on service and resource level react by sending intent reports to the respective intent owners one level up. This means that the business level intent manager receives an intent report corresponding to the service intent (S1) It can further report the degradation to the customer in an intent report related to the business level intent (B1).

The service level intent manager is receiving an intent report from the resource level related to the resource intent (R1). This is an additional indication of the issue, next to the knowledge created by the service assurance.

The example issue in the catalyst use case is related to the latency requirement of the connectivity service. The service level intent manager's cognitive loop enters the phase of proposing and evaluating a solution. It selects a strategy of increasing priority for the service as countermeasure to the latency issue. From the perspective of the service intent layer this solution is implemented through an update of the resource intent. The orchestrator executes the detail based on a service specific to this type of healing action.

The resource layer intent manager reacts to the changed resource intent and adapts the slice configuration. After this modification the new observations show that the issue is fixed and the knowledge about it is cleared. This leads to reports on all levels indicating system compliance again.

3.3. Intent Example 1: Business Intent for Ordering Connectivity via Dynamic Pricing

3.4. Intent Expression (English)

"I want **connectivity** between my premises (**location A**) and **Data Center B**.

I am willing to pay 1**0$/hour per 1Gbps.**

Valid **asap** but ok with waiting until valid"

3.5. Intent Expression using TIO (Turtle)

The intent of this request, known as the Spot Price Bid Request, is for a customer of PSE (Price Sensitive Enterprise) to order a specific capability from the business layer. This is known as the "B1" intent in the IDAN Catalyst Phase 2 (B for Business). A capability is a general product that may be provided by multiple service providers, and it is the responsibility of the capability marketplace to select the appropriate product from one of these providers. This intent includes specific requirements for the chosen service or product, such as functional requirements for connectivity between two sites and non-functional requirements for a minimum bandwidth of 1 GBit/s and a price not exceeding 10 USD/h. The customer is open to a delayed delivery of the capability, and an intent report will be provided to keep the customer informed of the handling state and current parameters. The intent acceptance and rejection control extension model may be used to override certain rejection reasons to ensure that the intent is not rejected solely because it cannot be fulfilled immediately.

The initial code block is a series of prefixes for TIO and associated ontologies.

```
@prefix  rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix  icm:
<http://tio.models.tmforum.org/tio/v1.0.0/IntentCommonModel#> .
@prefix  imo:
<http://tio.models.tmforum.org/tio/v1.0.0/IntentManagementOntology#> .
@prefix  pbi:
<http://tio.models.tmforum.org/tio/v1.0.0/ProposalBestIntent#> .
@prefix  cat: <http://www.operator.com/Catalog#> .
@prefix spot: <http://www.sdo1.org/Models/DynamicpriceOntology#> .
@prefix  met: <http://www.sdo2.org/TelecomMetrics/Version_1.0#> .
@prefix    t: <http://www.w3.org/2006/time#> .
@prefix   ex: <http://www.example.org/IntentNamespace#> .
@prefix   iv: <https://tmforum.org/2021/07/intentValidity#> .
@prefix  itv: <https://tmforum.org/2021/07/intentTemporalValidity#> .
@prefix  cem:
<http://tio.labs.tmforum.org/tio/v1.0.0/CatalystExtensionModel#> .
@prefix  xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix  arc:
<http://tio.models.tmforum.org/tio/v1.0.0/IntentAcceptanceControl/> .
```

Code Block 2 Prefixes

```
ex:B1_PSE_Biz_Intent1
  icm:hasExpectation ex:B1E1_delivery ,
                     ex:B1E2_property ,
                     ex:B1E3_reporting ;
```

Code Block 3 Intent

Targets

The V2 intent common model introduced target objects to be used instead of blank nodes. The target specifies the collection of resources to be used, or a condition for selection that involves a set of resources to choose from. In this example, the target does not prescribe resources, but allows a selection of exactly one resource from the Collection of all resources.

To be compliant, the resource selected to be used needs to fulfill the Target condition as well as the requirements defined by expectations. The delivery expectation mainly steers the target resource selection, but practically, the intent handler would choose a resource that is overall preferential. This can mean it chooses a resource that is the cheapest one, which would allow us to meet all expectations.

The target used here specifies a condition for selectable resources. It tells that exactly one of the resources from a container can be selected. This deliberately leaves a choice to the intent handler. Here, the container to choose from is determined by the condition property icm:elementType. This means all individuals of the specified class are elements of that container. Here, the container contains all resources. This is the widest possible definition, because everything is a resource in rdf. This means that the target definition tells you to choose exactly one element from the container of all resources and use that to fulfill the expectations associated with this target.

The target itself is otherwise an empty container, because a choice of resources was not yet done when creating the intent. The intent report will however drop the condition and provide a concrete container of resources that were selected.

The second target is a container with the intent itself as single member, it does not leave any choice and just tells that the report shall be about the intent.

```
ex:T1_Capability
  a icm:IndividualTarget ;
  icm:OneElementOf [ icm:elementType rdf:Resource ] ;
.
ex:T2_B1_Reporting
  a icm:IndividualTarget ;
  rdfs:member ex:B1_PSE_Biz_Intent1 ;
```

Code Block 4 Targets

Delivery Expectation and Parameters

The delivery expectation is mainly concerned to specify that something needs to be delivered. Delivery usually involves providing access to a service, product or capability and this typically requires that resources need to be used to provide it. A delivery expectation is therefore concerned with two main questions:

1. What is the thing that needs to be delivered?
   This constitutes a functional requirement, because it is about the nature of the needed thing.

```
ex:B1E1_delivery
  a icm:DeliveryExpectation ;
  icm:target ex:T1_Capability ;
  icm:allOf [ rdfs:member ex:B1P1_delivery_type ;
              rdfs:member ex:B1P2_delivery_condition ;
            ] ;
.

# Select a resource (service, product) according to the description in the
# catalog. Furthermore the resource shall be an object of a class in the
# ontology.  . This showcases the flexible approach of using ontology with
a catalog approach
ex:B1P1_delivery_type
  a icm:Parameter ;
  icm:deliveryDescription cat:Connectivity_DynamicPrice ;
  icm:deliveryType ex:ConnectivityCapability_DynamicPrice ;
.

# Only deliver if target resources can be selected that would meet the
# condition. This means the observation is associated with the target.
# If no suitable target resource is available to make this observation,
the
# parameter would be evaluated as not compliant.
ex:B1P2_delivery_condition
  a icm:Parameter ;
  icm:observation [ icm:latestValueOf spot:Price ] ;
  icm:atMost [ icm:value 10 ;
               spot:unit spot:unitDollarHour ;
             ] ;
```

Code Block 5 DeliveryExpectations

Note that a ***Proposal Expectation*** is not needed here, as the regular reporting already provides the information needed about current price.

Property Expectation

A property expectation always defines requirements based on observable properties of the system, setting goal/target values and thresholds for these properties. The vocabulary of the intent common model allows to express this by defining the condition under which the system would be considered compliant. The condition can be based on individual target values versus an observation, and it can be based on collection/set logic. This example defines three property-based requirements, and all of them need to be fulfilled to consider the system compliant to the property expectation.

```
ex:B1E2_property
  a icm:PropertyExpectation ;
  icm:target ex:T1_Capability ;
  icm:allOf [ rdfs:member ex:B1P3_price ;
              rdfs:member ex:B1P4_bandwidth ;
              rdfs:member ex:B1P5_location ;
            ] ;
```

Code Block 6 PropertyExpectation

Price based condition

The system is compliant if the price of the target is below 10 USD per hour. This requirement might be redundant here, because it is already a delivery condition. A non-compliant delivery condition would stop the delivery. The same condition within a property expectation would not imply if it shall be delivered or not. It would only indicate that there might be a problem with the delivered thing. So there is a semantic difference.

```
ex:B1P3_price
  a icm:Parameter ;
  icm:observation [ icm:latestValueOf spot:Price ] ;
  icm:atMost [ icm:value 10 ;
               spot:unit spot:unitDollarHour ;
             ] ;
.
```

Code Block 7 price condition

Bandwidth Condition

The wanted connectivity capability shall allow usage of at least 1 GBit/s. As the used metric measures the overall bandwidth across all users, the requirement is also about the accumulated available bandwidth. Please note that this requirement would only fail if there is actually demand above 1Gbit/s. This is implied by using an available Bandwidth metric, rather than a current bandwidth. unit80000 uses a unit string according to ISO/IEC 80000 standard.

```
ex:B1P4_bandwidth
  a icm:Parameter ;
  icm:observation [ icm:latestValueOf met:AvailableBandwidth ] ;
  icm:atLeast [ icm:value 1 ;
                icm:unit80000 "'GBit/s'" ;
              ] ;
.
```

Code Block 8 Bandwidth Condition

Location Condition

There is a collection of all connected locations. This compliance condition # states that the set of all locations that shall be connected is included in # the set of all connected locations.

```
ex:B1P5_location
  a icm:Parameter ;
  icm:subjectContainer [ rdf:member "'locationA'" ;
                         rdf:member "'locationB'" ;
                       ] ;
  icm:includedIn met:ConnectedLocations ;
.
```

Code Block 9 Location Condition

Reporting Expectation

Ask for reporting about key events and regular reports every 10 minutes. The scope of the reporting is the entire intent as indicated by the target. The receiver of the report is the intent management Function of Customer1. It is represented by the resource ex:IMF_Customer1. icm:NewEventTypes is a container of types (classes) of the events that were observed since the last report. A report shall be created if this container intersects with the container of those event types a report is wanted for. Compliance is reached if the report is created and received. icm:reportingInterval is a context. It sets a timer for interval-based reporting. This timer will generate an event of class icm:reportingIntervalExpired. Multiple of these timers can be specified, but here we only use one. Compliance is reached if the event is correctly issued. Send a report about the intent to the intent owner if since last report any of these events were observed. Also set a timer for regular reporting.

```
ex:B1E3_reporting
  a icm:ReportingExpectation ;
  icm:target ex:T2_B1_reporting ;
  icm:receiver ex:IMF_Customer1 ;
  icm:allOf [ rdfs:member ex:B1P6_reporting ;
            ] ;
  icm:reportingInterval [ a t:Duration ;
                          t:numericDuration "'10'" ;
                          t:temporalUnit t:unitMinute ;
                        ] ;
.

ex:B1P6_reporting
  a icm:Parameter ;
  icm:subjectContainer icm:NewEventTypes ;
  icm:intersectWith [ rdf:li icm:ReportingIntervalExpired ;
                      rdf:li icm:IntentRejected ;
                      rdf:li icm:StateComplies ;
                      rdf:li icm:StateDegrades ;
                      rdf:li icm:HandlingEnded ;
                      rdf:li icm:UpdateRejected ;
                      rdf:li icm:UpdateFinished  ;
                    ] ;
.
```

Code Block 10 ReportingExpectation

3.5.1. Explanation of Code Example : See B1_catalyst_business_intent.ttl

As with any example, this Turtle code defines a set of resources and their relationships in an RDF graph. The code uses several prefixes to abbreviate URI references, including **rdf:**, **rdfs:**, **icm:**, **imo:**, **pbi:**, **cat:**, **spot:**, **met:**, **t:**, **ex:**, **iv:**, **itv:**, **cem:**, **xsd:**, and **arc:**.

Expectation

The code defines an **icm:Intent** resource named **ex:B1_PSE_Biz_Intent1**. This intent represents a Spot Price Bid Request from a Price Sensitive Enterprise (PSE) customer to the ***business layer***. It is the B1 intent in the IDAN Catalyst Phase 2.

The PSE customer wants to order a capability, which is an abstract product that can be delivered by a capability marketplace choosing an appropriate product from any of the associated service providers. This intent specifies the requirements for choosing a service/product from a service provider that would deliver the desired capability, including both functional and non-functional requirements.

The functional requirement is that connectivity should be provided between two sites. The non-functional requirements are that a minimum bandwidth of 1 GBit/s should be available and that the price should not exceed $10/hour.

Key Condition of this Expectation

We now analyze this requirements more closely in this key section (See the code example from lines 170 through 176).

This section of the Turtle code defines a resource named **ex:B1P2_delivery_condition**, which is an **icm:Parameter**.

The **icm:observation** property of this parameter specifies that the latest value of the **spot:Price** property should be observed. The **icm:atMost** property specifies that the observed value should be at most 10 units of **spot:unitDollarHour**.

This means that the parameter is evaluated as not compliant if there is no suitable target resource available to make the observation, or if the observed value of the **spot:Price** property is greater than 10 units of **spot:unitDollarHour**.

The intent has several properties, including a **rdfs:comment** property that provides a description of the intent, an **imo:intentOwner** property that specifies the owner of the intent (in this case, the **ex:PSE_Customer** resource), a **cem:layer** property that specifies the layer at which the intent is targeted, and some **arc:intentRejectionReasonOverride** and **arc:updateRejectionReasonOverride** properties that specify certain rejection reasons that should be overridden.

The intent also has several **icm:hasExpectation** properties that link to other resources representing expectations for the intent. These expectations include the **ex:B1E1_delivery** expectation, which specifies the desired delivery of the capability, the **ex:B1E2_property** expectation, which specifies the desired properties of the capability, and the **ex:B1E3_reporting** expectation, which specifies the desire for reports about the current handling state of the intent.

ReportingExpectation

The `ex:B1E3_reporting` expectation, or ReportingExpectation, is expressed in the Turtle code.

The `ex:B1E3_reporting` resource is defined as an `icm:Expectation`, and it has several properties that specify details about the expectation.

The `icm:type` property specifies that the expectation is a `imo:ReportingExpectation`, which means that the customer expects to receive reports about the current handling state of the intent.

The `imo:validityPeriod` property specifies a time interval during which the expectation is valid. This is represented as an `imo:Interval` resource with a `t:hasBeginning` and a `t:hasEnd` property.

The `imo:hasReport` property links to a `imo:IntentReport` resource, which represents a report about the intent. The `imo:IntentReport` resource has several properties that provide information about the report, including the `imo:reportTime` at which the report was generated and the `imo:reportStatus` of the intent at that time.

Other

The code also defines several **icm:Target** resources, including **ex:T1_PSE_Site1**, **ex:T2_PSE_Site2**, and **ex:T3_PSE_Capability**, which represent the two sites and the desired capability, respectively. These resources have various properties that provide

more information about the targets, such as their **icm:type** and their **imo:validityPeriod**.

This example also has several other properties, including a **rdfs:comment**, an **imo:intentOwner**, a **cem:layer**, and some **arc:intentRejectionReasonOverride** and **arc:updateRejectionReasonOverride** properties. The resource also has several **icm:hasExpectation** properties that link to other resources.

The code also defines several other resources, including **ex:B1E1_delivery**, **ex:B1E2_property**, and **ex:B1E3_reporting**, which are expectations for the intent. These resources have various properties that provide more information about the expectations, such as the **icm:type** of the expectation and the **imo:validityPeriod** of the expectation.

The code also defines several other resources, including **ex:T1_PSE_Site1**, **ex:T2_PSE_Site2**, and **ex:T3_PSE_Capability**, which represent targets for the intent. These resources have various properties that provide more information about the targets, such as their **icm:type** and their **imo:validityPeriod**.

Summary

1. There is an **icm:Intent** named **ex:B1_PSE_Biz_Intent1** that represents a Spot Price Bid Request from a Price Sensitive Enterprise (PSE) customer to the business layer.
2. The PSE customer wants to order a capability, which is an abstract product that can be delivered by a capability marketplace choosing an appropriate product from any of the associated service providers.
3. The intent has a **rdfs:comment** property that provides a description of the intent, an **imo:intentOwner** property that specifies the owner of the intent (in this case, the **ex:PSE_Customer** resource), a **cem:layer** property that specifies the layer at which the intent is targeted, and some **arc:intentRejectionReasonOverride** and **arc:updateRejectionReasonOverride** properties that specify certain rejection reasons that should be overridden.
4. The intent has several **icm:hasExpectation** properties that link to other resources representing expectations for the intent. These expectations include a **imo:DeliveryExpectation** named **ex:B1E1_delivery** and a **imo:ReportingExpectation** named **ex:B1E3_reporting**.
5. The **ex:B1E1_delivery** expectation has a **imo:validityPeriod** property that specifies a time interval during which the expectation is valid.
6. The **ex:B1E3_reporting** expectation has an **imo:hasReport** property that links to an **imo:IntentReport** resource, which represents a report about the intent. The **imo:IntentReport** resource has a **imo:reportTime** property that specifies when the report was generated and a **imo:reportStatus** property that specifies the status of the intent at that time.
7. The code defines several **icm:Target** resources, including **ex:T1_PSE_Site1**, **ex:T2_PSE_Site2**, and **ex:T3_PSE_Capability**, which represent the two sites and the desired capability, respectively.

3.5.2. Requirements and Constraints

Description of Intent-driven Interface Integration Solution, including intent requirements, parameters

4. Appendix A: Abbreviations Used within this Document

4.1. Abbreviations & Acronyms

Abbreviation/ Acroynm	Abbreviation/Acronym Spelled Out	Definition	Source
TIO	TM Forum Intent Ontology		IG1258 Autonomous Networks Glossary
IDAN	Intent-driven Autonomous Networks		Name of the TM Forum Catalyst Project maturing intent work within TM Forum

5. References

Reference	Description	Source	Brief Use Summary
TIO	TM Forum Intent Ontology	IG1253, TR292	Links to formal Intent guides
Link to IDAN Catalyst code examples	https://github.com/intent-driven/TMF921-reference-implementation/tree/main/ontologies	Github	The catalyst examples were contributed by the IDAN phase 2 catalyst team

6. Appendix B: Detailed Intent Examples

6.1. Business Intent for Ordering Connectivity via Dynamic Pricing

See file source at B1_catalyst_business_intent.ttl at https://github.com/intent-driven/TMF921-reference-implementation/blob/main/ontologies/B1_catalyst_business_intent.ttl

```
@prefix  rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
@prefix  icm:
<http://tio.models.tmforum.org/tio/v1.0.0/IntentCommonModel#> .
@prefix  imo:
<http://tio.models.tmforum.org/tio/v1.0.0/IntentManagementOntology#> .
@prefix  pbi:
<http://tio.models.tmforum.org/tio/v1.0.0/ProposalBestIntent#> .
@prefix  cat: <http://www.operator.com/Catalog#> .
@prefix spot: <http://www.sdo1.org/Models/DynamicpriceOntology#> .
@prefix  met: <http://www.sdo2.org/TelecomMetrics/Version_1.0#> .
@prefix    t: <http://www.w3.org/2006/time#> .
@prefix   ex: <http://www.example.org/IntentNamespace#> .
@prefix   iv: <https://tmforum.org/2021/07/intentValidity#> .
@prefix  itv: <https://tmforum.org/2021/07/intentTemporalValidity#> .
@prefix  cem:
<http://tio.labs.tmforum.org/tio/v1.0.0/CatalystExtensionModel#> .
@prefix  xsd: <http://www.w3.org/2001/XMLSchema#> .
@prefix  arc:
<http://tio.models.tmforum.org/tio/v1.0.0/IntentAcceptanceControl/> .

##### Intent
#####################################################################
#
#  This intent represents the Spot Price Bid Request from PSE (Price
Sentitive
#  Enterprize) Customer to the business layer.
#  It is the B1 intent in the IDAN Catalyst Phase 2
#
#  The PSE customer orders a capability to be delivered. A capability is
an
#  abstract product that is not directly mapped to the product offering of
a
#  single service provider. It can rather be delivered by the capability
#  marketplace choosing an appropriate product from any of the associated
#  service providers.
#
#  This intent provides the requirements for choosing a service/product
from
#  a service provider that would deliver the wanted capability with
respect to
#  functional and non-functional requirements:
#
#  Functional requirement:
#       -> Connectivity shall be provided between two sites
#
#  Non Functional requirements:
#       -> A minimum bandwidth of 1 GBit/s shall be available
#       -> The price shall not exceed 10 USD/h
#
#  Intent reports would keep the intent owner informed about current
handling
#  state. This includes current values for the property parameters
regarding
#  price and available bandwidth.
#
#  The customer is ok with the capability not delivered immediately. This
#  means the customer is fine with a degraded intent. In order to emsure
#  that the intent handler does not reject the itnent, becasue it cannot
#  immediately fulfill it, the intent acceptence and rejection control
#  extension model is used to override certain rejection reasons.
#
ex:B1_PSE_Biz_Intent1
  a icm:Intent ;
  rdfs:comment "'Intent for ordering connectivity service via dynamic
pricing'" ;
```

```
  imo:intentOwner ex:PSE_Customer ;
  cem:layer "'business'" ;
  arc:intentRejectionReasonOverride imo:SuccessfulHandlingNotExpected ;
  arc:updateRejectionReasonOverride imo:SuccessfulHandlingNotExpected ;
  icm:hasExpectation ex:B1E1_delivery ,
                     ex:B1E2_property ,
                     ex:B1E3_reporting ;
.

##### Targets
#################################################################
#
# The latest intent common model introduced target objects to be used
instead
# of blank nodes. The target specifies the collection of resources to be
used,
# or a condition for selection that involves a set of resources to choose
from.
# In this example the target does not prescribe resources, but allows a
# selection of exactly one resource from the Collection of all resources.
#
# To be compliant the resource selected to be used needs to fulfill the
# Target condition as well as the requirements defined by expectations.
# The delivery expectation mainly steers the target resource selection,
but
# practically the intent handler would choose a resource that is overall
# preferential. This can mean it chooses a resource that is the cheapest
one,
# which would allow to meet all expectations.
#
# The target used here specifies a condition for selectable resources.
# It tells that exactly one of the resources from a container can be
selected.
# Thisdeliberately leaves a choice to the intent handler.
# Here, the container to choose from is determined by the condition
property
# icm:elementType. This means all individuals of the specified class are
# elements of that container. Here, the container contains all resources.
# This is the widest possible defintion, because everything is a resource
in
# rdf.
# This means that the target defintion tells to choose exactly one element
# from the container of all resources and use that to fulfill the
expextations
# associated with this target.
#
# The target itself is otherwise an empty container, because a choice of
# resources was not yet done when creating the intent. The intent report
# will however drop the condition and provide a concrete container of
# resources that were selected.
#
# The second target is a container with the intent itself as single member
# it does not leave any choice and just tells that the report shall be
# about the intent.
#
ex:T1_Capability
  a icm:IndividualTarget ;
  icm:OneElementOf [ icm:elementType rdf:Resource ] ;
.
ex:T2_B1_Reporting
  a icm:IndividualTarget ;
  rdfs:member ex:B1_PSE_Biz_Intent1 ;
.

##### Delivery Expectation and Parameters
#####################################
```

```
#
# The delivery expectation is mainly concerned to specify that something
needs
# to be delivered. Delivery usually involves providing access to a
service,
# product or capability and this typically requires that resources need to
be
# used to provide it.
#
# A delivery expectation is therefore concerned with two main questions:
#
# 1. What is the thing that needs to be delivered?
#    This constitutes a functional requirement, becasue it is about the
#    nature of the needed thing.
#    -> Catalog Based.  The intent common model allows to refer to an
external
#                       description of that thing. This can, for example,
be
#                       a service description in a catalog.
#    -> Ontology Based. The second way to define the nature of the thing
to
#                       be delivered is by requireing that this thing is
an
#                       object of a particular class within the ontology.
#
# 2. Under which conditions shall the delivery happen?
#    The delivery might be conditional and therefore the thing shall only
be
#    delivered if a given condition holds.
#
# The delivery expectaion in this intent refers to a capability
description
# in the marketplace catalog for specifying what needs to be delivered.
This
# intent also uses a type/class based specification of what shall be
deliverd.
# This means we are demonstrating a combined a catalog and ontology based
# approach.
#
# The delivery expectation practically steers the selection of target
# resources to be used to deliver the wanted thing. In this case the
target
# is an empty container, meaning there is not a fixed resource
# pre-determined to be used. Instead the target specifies the conditions
for
# valid resources.
#
# The system is compliant to a delivery expectation if the respective
# resources, which were selected to fulfill this delivery, meet the
# requirements of the delivery expectation. This means they comply to an
# external description of the thing to be delivered and/or they are
objects
# of a required class. Furthermore the condition for delivery needs to be
met
# for compliance.
#
ex:B1E1_delivery
  a icm:DeliveryExpectation ;
  icm:target ex:T1_Capability ;
  icm:allOf [ rdfs:member ex:B1P1_delivery_type ;
              rdfs:member ex:B1P2_delivery_condition ;
            ] ;
.

# Select a resource (service, product) according to the description in the
# catalog. Furthermore the resource shall be an object of a class in the
```

```
# ontology.  . This showcases the flexible approach of using ontology with
a catalog approach
ex:B1P1_delivery_type
  a icm:Parameter ;
  icm:deliveryDescription cat:Connectivity_DynamicPrice ;
  icm:deliveryType ex:ConnectivityCapability_DynamicPrice ;
.

# Only deliver if target resources can be selected that would meet the
# condition. This means the observation is associated with the target.
# If no suitable target resource is available to make this observation,
the
# parameter would be evaluated as not compliant.
ex:B1P2_delivery_condition
  a icm:Parameter ;
  icm:observation [ icm:latestValueOf spot:Price ] ;
  icm:atMost [ icm:value 10 ;
               spot:unit spot:unitDollarHour ;
             ] ;
.

##### Proposal Expectation
#####################################################
# Not needed, because the regular reporting would already provide the
# information needed about current price.


##### Property Expectation
#####################################################
# A property expectation always to define requirements based on observable
# properties of the system setting goal/target values and thresholds for
these
# properties.
# The vocabulary of the intent common model allws to express this by
defining
# the condition under which the system would be considered compliant.
# The condition can be based on individual target values versus an
observation,
# and it can be based on collection/set logic.
#
# This example defines three property based requirements and all of them
need
# to be fulfilled to consider the system compliant to the property
expectation.
#
ex:B1E2_property
  a icm:PropertyExpectation ;
  icm:target ex:T1_Capability ;
  icm:allOf [ rdfs:member ex:B1P3_price ;
              rdfs:member ex:B1P4_bandwidth ;
              rdfs:member ex:B1P5_location ;
            ] ;
.

# Price based condition. The system is compliant if the price of the
target
# is below 10 USD per hour.
# This requirement might be redundant here, because it is already a
delivery
# condition. A non compliant delivery condition would stop the delivery.
# The same condition within a property expectation would not imply if it
shall
# delivered or not. It would only indicate that there might be problem
with
# the delivered thing. So there is a semantic difference.
ex:B1P3_price
```

```
  a icm:Parameter ;
  icm:observation [ icm:latestValueOf spot:Price ] ;
  icm:atMost [ icm:value 10 ;
               spot:unit spot:unitDollarHour ;
             ] ;
.
# The wanted connectivity capability shall allow usage of at least 1
GBit/s
# As the used metric measures the overall bandwidth acorss all users, the
# requirement is also about the accumulated available bandwidth.
# Please note that this requirement would only fail if there is acutually
# demand above 1Gbit/s. This is implied by using an available Bandwidth
# metric, rather than a current bandwidth.
# unit80000 usues a unit string according to ISO/IEC 80000 standard
ex:B1P4_bandwidth
  a icm:Parameter ;
  icm:observation [ icm:latestValueOf met:AvailableBandwidth ] ;
  icm:atLeast [ icm:value 1 ;
                icm:unit80000 "'GBit/s'" ;
              ] ;
.
# There is an collection of all connected locations. This compliance
condition
# states that the set of all locations that shall be connected is included
in
# the set of all connected locations.
ex:B1P5_location
  a icm:Parameter ;
  icm:subjectContainer [ rdf:member "'locationA'" ;
                         rdf:member "'locationB'" ;
                       ] ;
  icm:includedIn met:ConnectedLocations ;
.

#### Reporting Expectation
#####################################################
#
# Ask for reporting about key events and regular reports every 10 minutes
#
# The scope of the reporting is the entire intent as indicated by the
target
#
# The receiver of the report is the intent management Function of
Customer1. It
# is represented by the resource ex:IMF_Customer1.
#
# icm:NewEventTypes is a container of types (classes) of the events that
# were observed since last report. A report shall be created if this
container
# intersects with the container of those event types a report is wanted
for.
# Compliance is reached if the report is created and received.
#
# icm:reportingInterval is a context. It sets a timer for interval based
# reporting. This timer will generate an event of class
# icm:reportingIntervalExpired
# Multiple of these timers can be specified, but here we only use one.
# Compliance is reached if the event is correctly issued.
#
# Speech Bubble: Sent a report about the intent to intent owner if since
last
# report any of these events were observed. Also set a timer for regular
# reporting.
#
ex:B1E3_reporting
  a icm:ReportingExpectation ;
```

```
    icm:target ex:T2_B1_reporting ;
    icm:receiver ex:IMF_Customer1 ;
    icm:allOf [ rdfs:member ex:B1P6_reporting ;
              ] ;
    icm:reportingInterval [ a t:Duration ;
                            t:numericDuration "'10'" ;
                            t:temporalUnit t:unitMinute ;
                          ] ;
  .

  ex:B1P6_reporting
    a icm:Parameter ;
    icm:subjectContainer icm:NewEventTypes ;
    icm:intersectWith [ rdf:li icm:ReportingIntervalExpired ;
                        rdf:li icm:IntentRejected ;
                        rdf:li icm:StateComplies ;
                        rdf:li icm:StateDegrades ;
                        rdf:li icm:HandlingEnded ;
                        rdf:li icm:UpdateRejected ;
                        rdf:li icm:UpdateFinished  ;
                      ] ;
  .
```

Code Block 11 Intent for ordering connectivity service via dynamic pricing

7. Administrative Appendix

7.1. Document History

7.1.1. Version History

Version Number	Date Modified	Modified by:	Description of changes
1.0.0	18-Aug-2023	Alan Pope	Initial Release

7.1.2. Release History

Release Status	Date Modified	Modified by:	Description of changes
Pre-production	18-Aug-2023	Alan Pope	Final edits prior to publication
Pre-production	18-Sep-2023	Adrienne Walcott	Updated to Member Evaluated status

7.2. Acknowledgments

This document was prepared by the members of the TM Forum Autonomous Networks Technical Architecture team.

Team Member (@mention)	Title	Company	Role*
Kevin McDonnell	Senior Director, Intelligent Automation	Huawei	Project Chair
Jörg Niemöller	Expert in Analytics and CEM	Ericsson	Project Chair
Yuval Stein	Director of Products Technologies, Solutions & AI	Amdocs	Project Chair
Kevin McDonnell	Senior Director, Intelligent Automation	Huawei	Author
yuan xie	Systems Expert	Huawei	Author
Jörg Niemöller	Expert of Analytics and Customer Experience	Ericsson	Key Contributor
Fernando Camacho	Product Director	Huawei	Key Contributor
Mohammed Rehan Reza	Researcher	Ericsson	Key Contributor
Lester Thomas	Head of Digital & IT New Technologies & Innovation	Vodafone	Key Contributor
Knut Johannessen	Chief Architect	Knut Johannessen	Key Contributor
guangying zheng	Autonomous Driving Network Standard Expert	Huawei	Key Contributor
Yuval Stein	Director of Products Technologies, Solutions & AI	Amdocs	Key Contributor

Team Member (@mention)	Title	Company	Role*
Aaron Boasman-Patel	Vice President, AI, Labs & Innovation	TM Forum	Additional Input
Alan Pope	Program Director	TM Forum	Additional Input
Dave Milham	Chief Architect	TM Forum	Additional Input
Xiao Hongmei	Pre-sales	Inspur	Reviewer