TELECOMMUNICATIONS PRODUCT DEFINING AND PROVISIONING

BACKGROUND

[001] In the past, telecommunications services were provided to customers by single entities, e.g., Telecom in Australia, or AT&T in the USA, or Post Office Telecommunications in the UK. These single entities defined and provided most or all of the technology required to provide a service, e.g., a telephone connection.

[002] In recent times, however, telecommunications products have become increasingly complicated, including a plurality of different and separated product components: (1) a physical communications network, (2) data transport or networking equipment, (3) applications that communication with customers to provide services, and (4) cloud infrastructure that is flexibly accessed through a network, e.g., platforms for the applications. Each product component may be configured in many different ways. For any given product, it may be necessary to ensure that the different product components perform within certain pre-defined parameters (e.g., network speed for certain applications, etc.).

[003] As a consequence of this recent, and on-going, splintering of the components for each telecommunications product, there is an increasing need for tools that allow telecommunications products to be defined and provisioned rapidly, efficiently and reliably.

[004] It is generally desirable to overcome or ameliorate one or more of the above described difficulties or to at least provide one or more useful alternatives.

SUMMARY

In accordance with the present invention, there is provided a computer-implemented process for defining a telecommunications product in accordance with a predefined data schema, the process including the steps of:

receiving, via a user interface, a request from a user for a specification for the telecommunications product;

receiving, via the user interface, characteristics data representing one or more characteristics for the specification from the user, including a plurality of properties for at least one of the characteristics;

receiving, via the user interface, input rules data representing at least one rule for the specification, including:

a selection of at least one of the received characteristics to provide a condition characteristic for the rule to operate,

a selection of a rule type to define operation of the rule, and

a selection of at least one of the received characteristics to provide a target characteristic on which the rule will operate;

processing the characteristics data and the input rules data, according to the predefined data schema, to generate specification data representing the characteristics in a specification document including a plurality of fillable data fields, and output rules data, in a rules document representing the rules for filling the fillable data fields,

such that the specification document and the rules document are useable together to:

generate order interface data, representing an order interface for a customer to request the product, including one or more of the properties by filling data into fillable data fields,

generate instance data representing an instance document including the data filled into the fillable data fields based on user input from the customer and data generated by processing the rules document in a rules engine, wherein the instance data, when re-combined with the specification document, represent product components of the requested product for provisioning the product.

[005] The present invention also provides process for provisioning a telecommunications product, including the process above and the steps of:

generating the order interface data; and

generating the product component data representing the product components.

[006] The present invention also provides a process including steps of:

generating specification data representing one or more specification documents, each including a plurality of fillable data fields, and rules data representing an associated rules document including rules for filling the fillable data fields, using user input from a user;

storing the specification data separately from the rules data; and

generating instance data representing an instance document including data filled into the fillable data fields based on user input and data generated by processing the rules in a rules engine.

[007] The present invention also provides a provisioning system having components configured to execute the processes above.

[008] The present invention also provides non-transitory storage with computer-readable instructions configured to implement the processes above.

[009] In accordance with the present invention, there is also provided a system, including:

a composer module that:

receives, via a user interface, a request from a user for a specification for the telecommunications product,

receives, via the user interface, characteristics data representing one or more characteristics for the specification from the user, including a plurality of properties for at least one of the characteristics,

receives, via the user interface, input rules data representing at least one rule for the specification, including:

a selection of at least one of the received characteristics to provide a condition characteristic for the rule to operate;

a selection of a rule type to define operation of the rule; and

a selection of at least one of the received characteristics to provide a

target characteristic on which the rule will operate, and

processes the characteristics data and the input rules data, according to the predefined data schema, to generate specification data representing the characteristics in a specification document including a plurality of fillable data fields, and output rules data, in a rules document representing the rules for filling the fillable data fields,

such that the specification document and the rules document are useable together to:

generate order interface data, representing an order interface for a customer to request the product, including one or more of the properties by filling data into the fillable data fields, and

a rendering engine, including a rules engine, that generates instance data representing an instance document including data filled into the fillable data fields based on user input from the customer and data generated by processing the rules document in the rules engine, wherein the instance data, when re-combined with the specification document, represent product components of the requested product for provisioning the product.

BRIEF DESCRIPTION OF THE FIGURES

[010] Preferred embodiments of the present invention are hereinafter described, by way of example only, with reference to the accompanying drawings, in which:

[011] Figures 1A and 1B are schematic diagrams of a product defining and provisioning system for telecommunications products;

[012] Figure 2 is a schematic diagram of hardware components of the system;

[013] Figure 3A is a data model diagram for a data schema of the system;

[014] Figure 3B is a block diagram of data structures in the system;

[015] Figures 4A and 4B is a flow chart showing steps in a defining and provisioning process performed by the system;

[016] Figure 5 is a sequence diagram of a create-customer-order process performed by the system;

[017] Figures 6A and 6B is a sequence diagram of an update-customer-order process performed by the system;

[018] Figures 7 – 11 are screen shots of a design user interface of the product provisioning system;

[019] Figure 12 is a screen shot of an order user interface of the product provisioning system; and

[020] Figure 13 is a block diagram of a computer of the product provisioning system.

DETAILED DESCRIPTION OF PREFERRED EMBODIMENTS OF THE INVENTION

Overview

[021] Herein is described a system and a process for defining and generating a telecommunications product offering, and ordering and provisioning the telecommunications product using the offering, in accordance with a predefined data schema. The system and process allow a telecommunications offering designer (a user who may be a product design engineer or product manager, e.g., a person working for a telecommunications company), to generate a schema-compliant machine-readable specification that defines an offering from the telecommunications company (e.g., a bundle of existing products and/or services) without having to manually write or edit machine-readable code (e.g., extensible markup language (XML) files). The system and process also allow the designer to generate a schema-compliant machine-readable ruleset for each specification, again without having to manually write or edit machine-readable code. The ruleset includes rules that define associations between properties of the specification that are automatically applied to simplify and verify data instances generated using the specification. The ruleset represents customisable logic that operates on properties of the specification based on input from the properties of the specification: the rules may generate outputs for the same properties from which they receive inputs, and/or for different properties. The specification and the ruleset are automatically applied by the system and process to generate schema-compliant machine-readable specification instances for orders based on input from submitters (e.g., users who are ordering one of the defined offerings, or users making orders on behalf of an end customer). The specification instances, when re-

combined with their specifications, define and are used to provision the actual orders: an order specification includes one or more product specifications, which define respective separable portions of each order such that the portions can be divided automatically by a machine for provisioning separable products (which may be "services", e.g., network connections, or "resources", e.g., hardware devices, in telecommunications business terminology) in the order instance (e.g., one portion of the specification may relate to a new telecommunications connection, and another portion of the specification may relate to a new telecommunications device for attaching to the telecommunications connection).

[022] The described process includes the steps of:

receiving, via a user interface, a request from a user for a specification for the telecommunications product;

receiving, via the user interface, characteristics data representing one or more characteristics for the specification from the user, including a plurality of properties for at least one of the characteristics;

receiving, via the user interface, input rules data representing at least one rule for the specification, including:

a selection of at least one of the received characteristics to provide a condition characteristic for the rule to operate,

a selection of a rule type to define operation of the rule, and

a selection of at least one of the received characteristics to provide a target characteristic on which the rule will operate; and

processing the characteristics data and the input rules data, according to the predefined schema, to generate specification data representing the characteristics, and output rules data, representing the rules, such that the characteristics data and the output rules data are useable together to:

generate order interface data, representing an order interface for a customer to request the product, including one or more of the properties, and

generate product component data representing product components of the requested product, based on the predefined schema, for provisioning the product.

[023] Code libraries can be used by a portal to dynamically render screens and implement all the validation within a browser for the customer. The browser can use JavaScript code to

retrieve the specification and the rules documents from a document data store, and can render the screen by interpreting the two documents together. A specification instance is captured once the fields are filled on the screen, conformant to the structure of the specification and the rules in the rules document. If there is data supplied prior to screen rendering, then the specification instance is pre-filled by the system. Validation, applying the rules, is performed in the browser (not the back end), improving the responsiveness and user experience.

[024] Messages received by the system can be validated through the use of the same code libraries, specification and rules documents. The received message is interpreted against the relevant rules and specification documents, and rejected with reason if it fails to comply with either. Configuring the documents once, and being able to dynamically apply them to both Web and B2B (API) dynamically, allows for improved efficiency.

System 100

[025] As shown in Figure 1A, a product defining and provisioning system 100, in embodiments of the present invention, includes:

[026] a design layer 102 (that operates during a "design time" of a defining and provisioning process 400) configured to receive input from the designer to define a telecommunications product offering, and to generate data representing one or more specifications ((including characteristics data and relationships data) and data representing one or more respective rulesets (including rules data), wherein the relevant specifications are identified in data representing an offering document that defines the product offering;

[027] a service layer 104 (that operates during a "run time" of the process 400), in communication with the design layer 102, and configured to receive product order data from the customer, representing an order of the product offering (the generated data is referred to as an "order instance" or "order document" of the order specification, and it includes, or is associated with, at least generated one product instance of the product specification(s) defined in the offering document), via a user interface that uses the specification and the ruleset (including the characteristics data, rules data, and relationships data) that define the product offering to generate a user interface representation, user interface functionality, and validation of the user inputs;

[028] an orchestration layer 106 (that also operates during the "run time"), in communication with the design layer 102 and the service layer 104, and configured to receive the specification and the ruleset (including the characteristics data, rules data, and relationships data) from the design layer 102, and the order data from the service layer 104, to validate orders and to generate product component data representing product components of the requested product, based on a predefined schema, for provisioning the product; and

[029] a plurality of connectors 108A, 108B, 108C (also referred to as "interface layers") that receive automatically selected portions of the instances according to the predefined schema, and generate data representing telecommunications components and configurations (including hardware network components 110A, virtual network components 110B, firewall components 110C, and/or cloud computing or storage 110D, etc.) that are required to provide (or "provision") the selected order to the customer.

[030] As shown in Figure 1B, the system 100 includes a plurality of functional modules. The user-interface portions of the design layer 102 and the service layer 104, i.e., the portions that provide observable outputs to, and receive electronic inputs from, the designer and the submitter, are provided by a user-interface (UI) module 112 of the system 100. The UI module 112 includes a rendering engine 114 that accesses and uses the specifications and respective rulesets to generate the UI and validate user input: this active UI may be in a form, e.g., a Web form. The rendering engine 114 (or "renderer") includes a rules engine that processes the ruleset during the run time as the instance is entered to apply the rules continuously to the data inputs: this includes ensuring that input is compliant with the specification and the rules, indicating which input data is required, and automatically editing / creating portions of the instance as input data is received for other portions (based on the rules).

[031] The UI module 112 includes a composer module 116 that accesses the user inputs to generate, edit, or compose the specifications and the rulesets. The UI module 112 includes a validation engine 118 that accesses generated instances, and uses respective specifications and rulesets to determine whether each instances complies with the corresponding specification and ruleset, and generates a resulting validation result (e.g., "complies", or "error" and an error location).

[032] The system 100 includes an application programming interface (API) 120 that allows the UI module 112 to communicate (by sending and receiving messages, including XML and/or JSON messages) with the other components of the system 100. The API 120 includes at least one customer system connector 122 that provides external systems with access to the API 120: for example, an external customer system may have interface components that communicate with the system 100 without using all functions of the UI 112, e.g., to generate, to access and/or to edit the various system data documents (including the specifications, the rulesets, the offerings, the catalogs, the orders, the lifecycles, etc.).

[033] The system 100 includes an enterprise service bus (ESB) 124 that passes the messages from and to the API 120, and communicatively connects the other modules of the system 100, in particular the back-end modules described hereinafter. The UI module 112 and the API 120 may be regarded as front-end modules that form a front end of the system 100, whereas the ESB 124 and other modules form a back end of the system 100. The ESB 124 includes a validation engine 126 that validates the instances (in the same way as the validation engine 118 in the UI module 112: in this way the engine 126 can be a copy of the engine 118), and that validates all data messages into and out of the back end. The messages passing through the ESB 124 are validated by the ESB 124 using the data schema, thus validating that the data documents in the messages are schema-compliant. The data schema can be stored as data, referred to as a common information model (COM), in a COM component in the ESB 124.

[034] The ESB 124 includes one or more external connectors 128 that allow for data communications with external systems (outside the system 100) that are still validated for schema compliancy by the ESB 124.

[035] The system 100 includes a plurality of the back-end modules, each communicatively connected by the ESB 124, including:

[036] a catalog module 130 for managing catalog data representing catalogs of pluralities of offerings;

[037] a specification manager 132, also known as a specification module, that supports creation of specifications, access to specifications, rendering of specifications, lifecycle management of specifications (in accordance with the associated lifecycle), data

integrity validation of specifications, and storing of the specification data separately from the corresponding rules data, e.g., in separate data structures or files, so they can be accessed and used separately;

[038] a locator module 134 for managing site-specific data representing customer sites;

[039] a customer order manager (COM) 136, also known as an order module, that manages creation and modification of orders;

[040] a process module 138 that receives and stores customer-specific delivery process documents, each including a delivery process definition (e.g., based on processes of the telecommunication provider, including details of installers) and a machine-readable identifier (ID) to associate the delivery process document with orders and instances; and

[041] an inventory manager (IVM) 140, also known as an inventory module, that manages inventory data. The components of the system 100 (including the modules 112 to 140, the design layer 102, the service layer 104, and the orchestration layer 106) are provided by at least one computer server 202 that communicates with a plurality of computing clients 204A, 204B, 204C over at least one data network 206, as shown in Figure 2. The components of the system 100 are stored as non-transitory computer-readable instructions in storage of the server 202, and served to the clients 204A, 204B, 204C, which may include providing data for a world-wide web browser of the clients 204A, 204B, 204C so that the user or the submitter can use the browser to access the front end (including the design layer 102 or the service layer 104), in particular the UI module 112. The UI module 112 includes portions that are served to the clients 204, including the rendering engine 114, composer 116 and validation engine 118, and portions that remain on the server 202 and communicate with the clients 204. The portions of the UI module 112 for the design layer 102 and the service layer 104 are served by the server 102 as instructions that are executed by clients 204 (including the browsers) to generate the user interfaces, and receive the data inputs (which may include data definition libraries and Java code being loaded into the browsers), so that only the documents, including the specifications (represented by the specification data including the characteristics data), the rulesets (represented by the rules data), the offerings and the orders and the catalogs, and the

lifecycles (represented by lifecycle data), need be transmitted over the network 206 between each client 204A, 204B, 204C and the server 202.

[042] The design layer 102 of the UI module 112 includes the composer component 116 that provides the user interface for the user to generate the system documents (including the specification data, the rules data, and the relationships data, etc.). The design layer 102 includes the rendering engine 114 (also referred to as a renderer component or a rendering component) that is used by the designer to view a rendering of the order-entry interface after the specification and the ruleset (including the specification data, the rules data, and the relationships data) have been generated by the composer component 116. The renderer component 114 may be downloaded from the server 202 to the client 204, and operate to process the entered data to render the order interface without sending or receiving data from the server 202, thus shifting the processing associated with rendering the order interface from the server 202 to the clients 204 (which may reduce network usage and server load, and may increase responsiveness for the designer user). The renderer component 114 and the composer component 116 are referred to as tools, or engines, for processing the user input data and the system document, including the specifications, the rules, the instances, etc. (represented by the specification data, the rules data, and the relationships data, etc.). In an example, the engines may be implemented using Java and/or Java Script.

[043] The service layer 104 of the UI module 112 accesses the renderer component 114 to render customer interfaces on the clients 204 to receive user-input data representing the orders (to generate the order instances and product instances) from the submitter. The service layer 104 accesses the back-end components in the product system 100 using the ESB 124 that provides a data gateway to the other back-end components.

[044] The service layer 104 includes the validation engine 118 of the UI module 112 that validates each specification instance against the ruleset (i.e., the rules data), and validates in turn each related specification in the relationships data, when an order is being made.

[045] The specification documents (represented by specification data and the relationships data), the rulesets (represented by the rule data), the offerings, the catalogs and the orders (represented by the specification data), and the lifecycles (represented by the lifecycle data)

are received as messages by the ESB 124 from the back end or from the front end or from the external systems. As described hereinbefore, the ESB 124 validates the messages using the schema. The schema is represented by schema definition data that in turn represents a data model 300A, as shown in Figure 3A, in which each product offering class can include a selectable plurality of characteristics (referred to as "SpecCharacteristics") and relationships (referred to as "SpecificationRelationships") for each specification (referred to as a "Specification"). The schema definition data may be formatted as an XML Schema Definition (XSD) file.

[046] As shown in Figure 3B, the system 100 stores and communicates the system data in data structures that are defined by, and conform to, the schema, including a specification document 302 for each specification, a ruleset document 304 for each ruleset (that is linked to the corresponding specification document 302 by an association including a specification ID or ruleset ID), an offering document 306 for each offering (including a plurality of specification IDs that identify respective specifications in the offering), a catalog document 308 for each catalog (including a plurality of offering IDs that identify respective offerings in the catalog), a lifecycle document 312 for each lifecycle (including a lifecycle ID that is used in corresponding specifications), an order document 310 for each order, and an instance document 312 for each instance (including an instance ID to identify the instance). The lifecycle document (e.g., an XML document) includes data representing a series of states that can be applied to an instance, and the stated represent stages of the instance that occur during the process 400, including a draft state, a submitted state, an acknowledged state, a rejected state, a complete state): the lifecycle is defined by a designer to accord with management process of a telecommunications provider, and can then be associated to a plurality of specifications, including orders, so that the rendering engine 114 can display the current state of an order or an instance using the current lifecycle state, can display past and future states, and can accept user input to progress to the next state defined in the associated lifecycle. Thus lifecycle management is associated with each order based on common lifecycles defined for a plurality of orders. The order document 310 is a specification for available orders, and includes order-specific characteristics, including order-termination conditions set by the designer. The order instance is associated with a plurality of product instances in the order that are generated based on input from by the submitter using the service layer 104.

[047] The specifications may be referred to as "dynamic-data specifications", and represent highly flexible documents for describing all data requiring dynamic typing and definition (including in relation to the product offerings, the customer orders, the services and resources in the specifications, etc.). The dynamic-data specifications facilitate rapid tooling/customisation while decoupling specification releases from operational releases to improve product time-to-market and reduce operational costs for provisioning telecommunications products. The dynamic-data specifications are structured, conform to industry standards, and are machine readable (and may be XML documents). The specifications are maintained by the specification manager 132 that supports specification creation, access/rendering, life cycle management, data integrity validation. The dynamic specifications may enable the adaptation and accommodation of these ongoing changes with minimal operational expenditure, and conversion into alternative formats to support external system integration. The specification may be referred to as a grouping construct which provides structure to the dynamic data patterns (Characteristic Specification / Characteristic Value) available within a standard format, e.g., the TMForum's Information Model (SID), set by the predefined schema. As described herein, the specification is used for applying structure to stored dynamic data (the instances), where a specification of a given type and version is used when writing the dynamic data, and the same specification type and version needs to be used to interpret the dynamic data when reading. The specification may also be used for providing a "contract" for dynamic data exchanges been systems, including between organisations, e.g., when sending order messages to the external systems. A message (the instance) is formed to be compliant to the specification and can then be interpreted or read using the specification as a template. The specification is used for rendering the user interfaces by the renderer component 114: the specification contains the base information required to render a user interface, and the data collected from the user interface forms the "specification instance". This specification instance can then be used within messages for exchange with other systems, which also utilised the same type and version of the specification. A Specification Instance is the data captured when using the specification as a template: for example a product specification provides a template for a telecommunications product (which may be associated with a “service” or a “resource”) represented by a product specification instance associated with an order instance. The specifications are used to render

screens, store captured data in inventory, and create and validate messages exchanged with external systems.

[048] Each specification (represented by the specification data) is generated (in the process 400) to include one or more characteristics (represented by characteristic data in the specification data), and each characteristic is generated to have one or more selected properties that define how the characteristic is used: when a specification is rendered by the render 114, the characteristics of that specification define fillable data fields to accept use input data, and the properties of the characteristic define what data values are acceptable in the data fields, including data types. The available properties and characteristics are defined by the schema. The characteristics and properties of each specification are selected by the designer (using the design layer 102), and the data is entered into the data fields of the specification by the submitter (using the service layer 104), limited by the properties of each characteristic (and assisted by the ruleset), to generate an instance.

[049] The relationships specify other specifications to which the current selected specification relates, and can associate different product specifications that relate to different provisioning entities into one order specification. In an example, a relationship is defined, by the designer, to associate two product specifications, where one specification is used to receive data from a submitter (to define a product and order) while the other specification is not made available to the submitter, and thus is not exposed to one of the clients 204 after the design time, but is includes data that is used during completion and provisioning of the order (e.g., data representing information used by the external provisioning systems, but not of relevance to the submitter—for example, this could be information on telecommunication suppliers who provision the actual telecommunications "services" and "resources" defined in the product instance).

[050] The characteristics may have one of the following selected types, defined in the schema: Enumeration Type, Key Value Type, Range Type, and Collection Type.

[051] The Enumeration Type provides a list of values to select from of a given type (e.g., String, Integer, Float, etc.) from which a single value may be selected. The enumerated list

may have optional properties applied including being made read only, having a default value, or corresponding to a unit of measure for each value.

[052] The Key Value Type includes Name and Value that conforms to given data type (e.g., String, Integer, Float, Date-Time, etc.). The Key Value may have optional properties applied including being made read only, having a default value, or corresponding to a unit of measure for each value.

[053] The Range Type is a Key Value that accepts a single numeric value within a constrained range. The Key Value may have optional properties applied including being made read only, having a default value, or corresponding to units of measure for each value.

[054] A characteristic of the Collection Type is a characteristic that enables other characteristics to be created as children and be grouped. For example, a Collection of User Details could be created with children characteristics of First Name, Last Name and Phone Number. The grouping formed with collections is used for a range of purposes including for presentation and to support rules, allowing a rule to be applied to a parent and apply to each of the child characteristics.

[055] The characteristics may have one or more of the following properties.

[056] The Characteristic Property (also referred to as a characteristic option) controls the isolated behaviour of the characteristic.

[057] The Cardinality Property controls the number of instances of the characteristic within the specification instance. The minimum cardinality value is the minimum number of characteristic instances required. For example, if the minimum cardinality is one then one instance must be provided in order for the specification instance to be compliant to the specification. Or another way of considering it is that in a field that providing a value is mandatory. A specification instance can have characteristic instances up to the maximum number specified in the cardinality instance property. This is useful in dealing with multiplicity relationships like number of logical or physical ports on a piece of network equipment for example.

[058] The Default Value Property is the default value of the characteristic that can be set automatically within the specification instance. The value of the characteristic is still able to be overwritten by the user.

[059] The Unit of Measure (UOM) Property is a user defined String that can be applied to provide further information on a characteristic and can be any character. For example a Unit of Measure can be applied to a price characteristic as "USD".

[060] The Read Only Property allows the characteristic to be defined and visible, but not editable by the user. For example, the price of a product may be displayed, but not editable by the user.

[061] The Encryption Property defines a public key, which is part of a public-key cryptography system, received from the designer using the UI module 112 during the design time. The Encryption Property is added to a characteristic that receives potentially confidential information, e.g., passwords. The public key of the Encryption Property is then used by the UI module 112, based on user input from the submitter during run time, to encrypt data entered into the corresponding characteristic (e.g., password text) in the client 204 of the submitter so that the data for this characteristic, stored in the generated instance, remains encrypted while in the system 100. The private key, corresponding to the public key, is used in an external system to decrypt the encrypted data once the encrypted data has left the system 100. In this way, certain data fields (characteristics) can be used to carry confidential data through the system 100.

[062] The rules, in the rules data, are applied to each product specification, and to each product offering, and to each order specification.

[063] The rules in the ruleset are used to manage the interactions between the characteristics of a specification. For example, a rule can be created to: (i) modify a characteristic once preset conditions are satisfied - e.g. automatically set the price, once a given product is selected in an Enumeration; (ii) add or remove a characteristic if preset conditions are satisfied - e.g. add additional values for the contact details of the building owner in a multi-dwelling unit; (iii) make a characteristic read-only if preset conditions are satisfied; or (iv) hide a characteristic from the user if preset conditions are satisfied. The rules may be used to implement company

specific business rules, apply constraints, reduce errors and improve the user experience in creating the specification instance. The format for rule representation between trading partners may be of a lower relevance than the format of specifications and their characteristics and properties.

[064] The rules extend specifications through a common set of definitions with which to describe inter characteristic behaviours. This means that much of the logic that may have previously been coded into Portals, Integration Layers, and Backend Components can, in the system 100, be described in commonly useable rules data for each specification. The rules are applied as an addition to the specification, and may not override or invalidate the specification itself. Conditions may be specified for a rule to operate on order data entered into the characteristics of a specification; otherwise a rule will always be applied. Both specifications and offerings may have separate and independent sets of rules. Specification-level external rules are limited in scope to elements only within the specification itself. These rules are always applicable regardless of whether the specification is alone or is part of an offering. Offering-level external rules are only relevant within the offering itself; their effects on affected specifications do not apply when the same specification is in another offering. The rules defined on this level should only be extra-specification rules (relating elements across different specifications), or intra-specification rules that are conditional upon being a part of the offering.

[065] The following rule types are captured in the external rules document complementing the product definition documents, and are defined in the schema: value availability; availability; value; comment; read only; cardinality constraint; cardinality lock; and action availability.

[066] The Value Availability rule specifies which values of the target characteristic are to be available. The remaining values are unavailable, unless declared available by alternative rules. Where multiple Conditional Value Availability rules are applicable, a value will be available as long as the condition for at least one relevant rule is satisfied. This rule accepts multiple values for the same target. A valid target is SpecCharacteristic. Examples include: Location value set to "Outdoor" available if "NTD Type" is "External"; and Speed plan/sub-plan

selected may both influence the "Interface Speeds" available. (In an example, Speed may be 2-1000Mbps if plan Fibre and sub-plan ABC is selected.)

[067] The Availability rule specifies the availability of a target characteristic or specification. If the target is unavailable, it may not be ordered for the entity instance. This rule primarily differs from the Read Only rule in that the affected characteristic or specification will be excluded from the subsequent entity, whereas the latter rule will retain its last assigned value. Use this rule with caution, as it is possible for this rule to violate specification's definition of mandatory elements. Valid target are SpecCharacteristic, and Specification. Examples include: Ask for an "Authorisation Code" when user selects "MTU" greater than "MTU Generic Max"; and "Building Owner Name" and "Building Owner Contact Number" are only available when "End User Location" = "MUC".

[068] The Value rule specifies that the value of the target characteristic is set to the rule's target value. Changing the value after this rule is applied is possible if the characteristic is not read only. A valid target is SpecCharacteristic. An example is: Default value of "Transparent VLAN" dependent on whether or not selected access link allows multiplexing.

[069] The Comment rule specifies that the specified comment is specified to be relevant to the target characteristic. This rule is mainly of value when the specification is rendered visually. Valid targets are SpecCharacteristic and Specification. An example: If "PSTN Phone Number" is not specified when ordering ADSL, show a note specifying that the customer should inform service provider as soon as possible when one is obtained.

[070] The Read Only rule specifies that the target characteristic may no longer have its value changed manually. It will retain its last assigned value, and may still have its value changed via subsequent Value rules. If the target is a specification, this rule will be applied to all of its characteristics. Valid targets are SpecCharacteristic and Specification.

[071] The Cardinality Constraint rule constrains the cardinality range of the target to a subset of the range specified in the specification. If the target previously had a minimum cardinality of 0, raising it to at least 1 will make the target mandatory. Valid targets are: SpecCharacteristic and Specification. In an example, "Building Owner Name" is mandatory if end user location is a multi-dwelling unit.

[072] For the Cardinality Lock rule, valid targets are SpecCharacteristic and Specification.

[073] For the Action Availability rule, a valid target is Specification.

[074] The system 100 provides definition and application of business and technical rules, granting designers an additional layer of control over how their specifications are consumed and processed. This approach may dramatically reduce the degree of tooling required to introduce new products, while also ensuring that validation and business rules are applied in a consistent manner across all components.

Process 400

[075] In operation, the system 100 performs or executes the process 400, which includes:

[076] a design process (executed by the design layer 102 in "design time") to design available lifecycles, specifications, offerings and catalogs;

[077] an ordering process (executed by the service layer 104 in "run time") to generate one or more product instances and at least one order instance;

[078] a provisioning process to provision the telecommunications products based on the order and the lifecycles.

[079] In the design process, the design layer 102 executes a computer-implemented process for defining a telecommunications product in accordance with a predefined schema, the process including the steps of:

[080] receiving, via a user interface, a request from the designer (a user, e.g., a telecommunications product engineer) for a specification (a product definition) for the telecommunications product;

[081] receiving, via the user interface, characteristics data representing one or more characteristics for the specification from the user, including a plurality of properties for at least one of the characteristics;

[082] receiving, via the user interface, input rules data representing at least one rule for the specification, including:

[083] a selection of at least one of the received characteristics to provide a condition characteristic (an input) for the rule to operate,

[084] a selection of a rule type to define operation of the rule, and

[085] a selection of at least one of the received characteristics to provide a target characteristic on which the rule will operate; and

[086] processing the characteristics data and the input rules data, according to the predefined schema, to generate specification data (e.g., in the extensible markup language, XML) representing the characteristics, and output rules data, representing the rules in a ruleset, such that the characteristics data and the output rules data are useable together to:

[087] generate order interface data, representing an order interface for a submitter to request the product, including one or more of the properties, and

[088] generate product component data representing product components of the requested product, based on the predefined schema, for provisioning the product.

[089] The design process includes receiving a render request from the designer to render an order interface. The render request is generated by clicking on a render button in the design interface.

[090] The design process includes receiving a rule with a rule name, rule description and rule type, and displaying the rule in the rules tab.

[091] The design process includes receiving characteristic name, a characteristic type, a cardinality, and a description, and displaying the characteristic name.

[092] The ordering process includes the renderer component using the specification data to render an order interface, optionally with selectable properties, showing the defined product, and allowing a submitter to enter data to order the new product.

[093] The ordering process includes receiving from the user a specification name, a specification type, a specification class, and a specification delivery process;

[094] The design process includes receiving a selection of at least one relationship for one of the characteristics, where the relationship specifies a different previously defined product specification, and displaying the relationship (in a relationships tab), and generating relationships data defining the relationships to form an offering.

[095] The design process includes optionally receiving field names and definitions (e.g., values in an enumerated list), depending on the characteristic type, and displaying these fields (in the specification tab).

[096] The ordering process includes validating data representing the instance against the specification data and the rules document.

[097] The provisioning process includes processing (including decomposing, or "breaking down") the specification, ruleset and instance, including the relationships in the order, to generate decomposed order data (in computer-readable files) for external systems of telecommunications service providers according to known standards, such as the TM Forum 'Frameworx' standards. The relationships define what other different sub-specifications need to be decomposed for the overall product to be provided, e.g., decomposed into "services" and "resources" by telecommunications suppliers.

[098] The design process may make it easier for a telecommunications provider to define new products, and generate order forms and instantiate the order data according to the specifications and rules, for customers.

[099] The provisioning process includes the step of

generating the product component data representing the product components.

[100] Also provided is non-transitory storage with computer-readable instructions configured to implement the processes above.

[101] As shown in Figures 4A and 4B, the process 400 includes at least the following steps:

[102] the UI module 112 receiving login data from the client 204 for the designer to log into the UI module 112, and loading a portion of the UI module 112 onto the client 204 from the server 202 (step 402);

[103] the UI module 112 receiving user input data to define and generate a lifecycle document (step 404);

[104] the UI module 112 generating an order-input interface, and receiving user input data to generate an order specification document (step 406);

[105] the UI module 112 receiving user input to generate a specification, or to load an existing specification, and loading a specification from the specification manager 132 to the client 204 (step 408);

[106] the UI module 112 receiving user input to generate initial properties for the specification (step 410), including a human-readable name and description, a machine-readable association (or "link") with a lifecycle document, a machine-readable schema class for the specification (in accordance with the schema) that defines a technical category of the specification in the schema, and a machine-readable association with a delivery process document stored in the process module 138;

[107] the UI module 112 receiving user input to generate one or more characteristics that are added to the specification document, each characteristic including properties (step 412), where the available characteristics and properties are defined in the schema as described hereinbefore, including a human-readable name, machine-readable characteristic types (described hereinbefore), machine-readable data type definitions for values (e.g., text, Boolean, integer, etc.)—the composer module 116 generates the specification (e.g., as an XML document) continuously as the user input is received;

[108] the UI module 112 receiving user input to generate relationships that define links to other specifications, and thus define a plurality of associated product specifications of an order specification (step 414);

[109] the UI module 112 sending the generated specification to the back end where it is validated by the ESB 124 according to the schema (step 416);

[110] the UI module 112 receiving user input to generate the ruleset, including receiving properties for each rule (step 418), selected based on the available rule types in the schema (described hereinbefore), and the ESB 124 accessing data from external systems using the external connectors 128 to access machine-readable properties for rules defined by external systems (e.g., specific for specific designers)—the ruleset is generated as a ruleset document (e.g., XML) as the rules are defined, and each ruleset has a defined one-to-one association with a previously generated specification by including a specification ID;

[111] the ESB 124 validating the generated ruleset based on the schema (step 420);

[112] the UI module 112 receiving user input to generate an offering (i.e., an offering document, e.g., in XML) that defines an association with one or more previously generated specifications, and defines one or more machine-readable qualification criteria (step 422);

[113] the UI module 112 receiving user input to generate a catalog (i.e., a catalog document, e.g., in XML) including associations with one or more of the previously defined offerings, and an association with a customer (e.g., a customer ID) defined in a customer database, e.g., a customer relationship management (CRM) database, which can be an external system accessed using the external connectors 128; and storing the catalog by the catalog module 130 (step 424);

[114] the UI module 112 receiving user input for the submitter (i.e., a person representing a customer) to log into the UI module 112, and receiving a selection of a customer, e.g., a customer ID, for the system 100 to determine which lifecycles, etc., are available to this customer based on the customer ID (step 426);

[115] the UI module 112 receiving user input to select a site location (e.g., a site ID) associated with the customer ID (step 428);

[116] the UI module 112 sending data representing the customer and the site location (e.g., customer ID and site ID) to an external system, e.g., through the external connectors 128, to determine which offerings are available for this customer at this site by automatically accessing data records based on the data representing the customer and the site location, and the system 100 receiving in return data representing offering eligibility (step 430);

[117] the catalog module 130 receiving data representing the offering eligibility, through the ESB 124, determining stored catalogs with qualification criteria that match the offering eligibility (thus the catalog module 130 selects eligible offers based on data identifying the customer and the site), and sending these catalogs to the UI module 112 (step 432);

[118] the UI module 112 generating a UI display, for the submitter, indicating the available catalog(s), and thus available offerings and specifications (step 434);

[119] the UI module 112 receiving user input from the submitter to select one of the available offerings (step 436);

[120] the UI module 112 loading the specifications, rules and lifecycle documents for the selected offering to the client 204, and the rendering engine 114 generating an active form for the submitted to view and enter data (step 438);

[121] the UI module 112 receiving user input to add data to the data fields of the characteristics of the loaded specification, and contemporaneously applying the rules in the loaded ruleset to control which use input is accepted by the UI module 112, and to automatically determine data for others of the data fields, using the rules engine, thus generating an instance of the order specification and one or more instances of loaded product specifications (step 440)—the data for each characteristic can be manually entered by the submitter, or automatically determined from the manually entered data and the rules (by the rules engine)—the UI module 112 receives user input to select the lifecycle state, which include selecting of the states in the lifecycle associated with each specification;

[122] the UI module 112 sending the order and product instances to the ESB 124, which validates them, including using the validation engine for the instances (step 442)—as the instance data is data received from a client 204, which may corrupted or non-compliant, the ESB 124 also re-validates the order specification, the product specification(s) and respective ruleset(s) as they are returned with the instances;

[123] the back end modules store the instances as instance data associated with the specifications (e.g., as values with respective characteristic IDs), thus multiple instances

associated with a specification can be stored relatively efficiently since only the data values are stored, not the data types, but the data types can still be determined for later validation, editing, etc. (step 444);

[124] the UI module 112 receiving user input data to load an existing instance into the UI module 112 for display, which includes loading the associated specification, but not the rules because stored instances have been validated (step 446);

[125] the UI module 112 receiving user input to edit an instance, which includes loading the instance, the associated specification and the associated rules into the UI module 112, displaying the instance, and operating the rules engine to validate data entry (step 448);

[126] the UI module 112 sending an instance to an order management system, which can be the COM 136 or an external system connected to the external connectors 128 (step 450);

[127] the UI module 112 receiving user input to load up the instance, and to progress to the next lifecycle state, which can include approving an order, or submitting an order, or invoicing an order, etc. (step 452);

[128] the UI module 112 receiving user input to complete or close an order (step 454), which is the step that effects provisioning, and the process module 138 separates different portions of the instance document that correspond to different portions (e.g., different departments in a the telecommunications organisation that provide different parts of an telecommunications product) based on identifiers in the instance that were defined in the specification;

[129] the system 100 storing completed instances in the IVM 140 (step 456);

[130] the UI module 112 receiving user input to access a stored product instance (step 458);

[131] the UI module 112 receiving user input to edit the accessed stored instance, again using the specification and the ruleset (step 460); and

[132] the UI module 112 again returning the edited instance to the COM 136 (step 462).

[133] The steps 402 to 424 form the design-time portion of the process 400. The steps 426 to 460 form the run-time portion of the process 400.

[134] To create a customer order, or to update a customer order, the system 100 performs a create customer order process 500 as shown in Figure 5, or an update customer order process 600 as shown in Figures 6A and 6B, respectively. In the processes 500 and 600, the UI component 501 (including the UI module 112) communicates with the ESB component 502 (including the ESB 124), which communicates with the COM component 504 (including the COM 136), and the COM component 504 communicates with the IVM component 506 (including the IVM 140). The "order" in these Figures refers to a data package that includes an order instance and at least one a specification instance.

[135] As shown in Figure 5, the create order process includes the steps of:

[136] the UI 501 sending an order to the ESB 502 (step 508);

[137] the ESB 502 sending a create customer order request to the COM 504 (step 510.0);

[138] the COM 504 setting the order status to ‘submitted’, and setting the interaction date (510.1), and persisting the order in the order database (510.2);

[139] in a loop, repeated for each order item that is changed, the COM 504 moving a party/place interaction to the inventory (510.3) and moving a sub item inventor to a managed entity relationship (510.4);

[140] to add or modify the order, sending create (510.5) or update (510.7) commands to the IVM 506, and receiving respective acknowledgements (510.6, 510.8);

[141] to delete an order, cancelling the order when it is complete;

[142] the COM 504 then progressing the order status to ‘acknowledged’ (501.9), updating the customer order status (501.10), generating a notification (510.11), and sending the notification to the ESB 502 (510.12).

[143] As shown in Figures 6A and 6B, the update customer order process 600 includes the steps of:

[144] the UI 501 sending an update request to the ESB 502 (step 512), which passes it on to the COM 504 (step 512.0), which accesses the original customer order (512.1), and return this to the UI 501 via the ESB 502 (513);

[145] the UI 501 sending an update request to the COM 504 via the ESB 502 (514), and the COM 504 updating the order (512.2);

[146] if the request contains inventory, the COM 504 sets the original item to inactive, and inserts a new item (512.3), moves party and place to the inventory level (512.4), and moves sub item inventory to managed entity relationship (512.5);

[147] in a loop for each active customer order item:

[148] to add an order item, the COM 504 checks with the IVM 506 for inventory existence (512.6, 512.7), and updates the inventory if it exists (512.8, 512.9), or creates the inventor if it does not exist (512.10, 512.11),

[149] to modify an order item, the COM 504 updates the IVM 506 (512.12, 512.13), and

[150] to remove an order item, the COM 504 cancels the inventory in the IVM 506 if the order is complete (512.14, 512.15);

[151] if the request does not contain inventory, in a loop for each customer order item, the COM 504 updates the inventory status in the IVM 506 (512.17, 512.18) or cancels the inventory in the IVM 506 is the order is complete (512.18, 512.19); and

[152] the COM 504 builds a customer order notification (512.20), and sends it to the UI 501 via the ESB 502 (512.21).As shown in Figures 7 and 8, the user interface 700

generated by the composer component 116 includes controls that allow a user to create a new specification. The specification and version is displayed in a first, left-hand column 702. When selected, the specification's contents (including the characteristics) are displayed in a second, middle column 704. When selected, a characteristic's properties are displayed for view, or for configuration, in a third, right-hand column 706. Properties of a characteristic are entered into the contents view of the specification (right-hand column), which may be under a specification tab 708.

User Interface (UI) Module 112

[153] The UI module 112 may provide a render control 710, as shown in Figure 7, that when selected, controls the rendering engine 114 to process the specification data, and rules data, and to generate and send data for a order interface, as shown in Figure 12, that includes the details of the available order as currently defined in the specification and rules. The render control 710 (which may be a button) allows a designer to rapidly switch between a design view of the product offering, and an input view as would be used by a customer to actually generate a specification instance. The order interface is provided in part by the renderer component 114.

[154] As shown in Figure 9, the relationships of a specification can be viewed and created using a relationships view in the right-hand column under a relationships tab 902 in the user interface provided by the UI module 112.

[155] As shown in Figures 10 and 11, the rules associated with a specification can be displayed and edited in the right-hand column under a rules tab 1002 in the user interface.

Computer Systems

[156] Figure 13 is a schematic block diagram of a computer system 1300 in the server 202, or in the client 204, or in both. The computer system 1300 may be based on a 32-bit or 64-bit Intel architecture, and the processes and/or methods executed or performed by the system 100 are implemented in the form of programming instructions of one or more software components or modules 1302 stored on non-volatile (e.g., hard disk) computer-readable storage 1304 associated with the computer system 1300. At least parts of the software modules

1302 could alternatively be implemented as one or more dedicated hardware components, such as application-specific integrated circuits (ASICs) and/or field programmable gate arrays (FPGAs).

[157] The computer system 1300 includes at least one or more of the following standard, commercially available, computer components, all interconnected by a bus 1316: random access memory (RAM) 1306, at least one computer processor 1308, and external computer interfaces. The external computer interfaces include: universal serial bus (USB) interfaces 1310 (at least one of which is connected to one or more user-interface devices, such as a keyboard, a pointing device (e.g., a mouse 1318 or touchpad), a network interface connector (NIC) 1312 which connects the computer system 1300 to a data communications network such as the Internet 1320, and a display adapter 1314, which is connected to a display device Q322 such as a liquid-crystal display (LCD) panel device.

[158] For the server 202, the computer system 1300 includes—in addition to the modules described hereinbefore (including with reference to Figure 1B) that implement the processes described hereinbefore (including the process 400)—a plurality of standard software modules, including: an operating system (OS) 1324 (e.g., Linux or Microsoft Windows); web server software 1326 (e.g., Apache, available at http://www.apache.org); database modules 1330 (e.g., MySQL, and MongoDB), which allow data to be stored in and retrieved/accessed from one or more databases 1332, and scripts and supporting data 1334, including markup language (e.g., HTML, XML) scripts, JavaScript, and/or CGI scripts, image files, style sheets, etc. The database modules include at least one relational database management system (RDBMS), e.g., MySQL, for storing data in a relational database (e.g., the selected rulesets, and "static" data), and at least one document-oriented database (or noSQL database), e.g., MongoDB, which can be faster and more accessible for storing the dynamic documents (e.g., the specification documents and instance documents).

[159] For the client 204, instead of a web server 1326, the computer system 1300 includes standard web browsing modules with scripting support (e.g., Google Chrome) to allow the design layer 102 to be downloaded to and executed at least in part as a web application in the computer system 1300.

[160] The boundaries between the modules and components in the software modules 1302 are exemplary, and alternative embodiments may merge modules or impose an alternative decomposition of functionality of modules. For example, the modules discussed herein may be decomposed into submodules to be executed as multiple computer processes, and, optionally, on multiple computers. Moreover, alternative embodiments may combine multiple instances of a particular module or submodule. Furthermore, the operations may be combined or the functionality of the operations may be distributed in additional operations in accordance with the invention. Alternatively, such actions may be embodied in the structure of circuitry that implements such functionality, such as the micro-code of a complex instruction set computer (CISC), reduced instruction set computer (RISC), firmware programmed into programmable or erasable/programmable devices, the configuration of a field- programmable gate array (FPGA), the design of a gate array or full-custom application-specific integrated circuit (ASIC), or the like.

[161] Each of the steps of the processes of the computer system 1300 may be executed by a module (of software modules 1302) or a portion of a module. The processes may be embodied in a machine-readable and/or computer-readable medium for configuring a computer system to execute the method. The software modules may be stored within and/or transmitted to a computer system memory to configure the computer system to perform the functions of the module.

[162] The computer system 1300 normally processes information according to a program (a list of internally stored instructions such as a particular application program and/or an operating system) and produces resultant output information via input/output (I/O) devices. A computer process typically includes an executing (running) program or portion of a program, current program values and state information, and the resources used by the operating system to manage the execution of the process. A parent process may spawn other, child processes to help perform the overall functionality of the parent process. Because the parent process specifically spawns the child processes to perform a portion of the overall functionality of the parent process, the functions performed by child processes (and grandchild processes, etc.) may sometimes be described as being performed by the parent process.

Example Documents

[163] An example specification rules document can include data representing the following XML:

```
[164] <SpecificationRuleset>
[165]     <ID>Bitstream 3 Rules</ID>
[166]     <Name>Bitstream 3 Rules</Name>
[167]     <Version>0.1</Version>
[168]     <AssociatedTo type="ProductSpecification">
[169]     <ID>Bitstream 3</ID>
[170]     <Name>Bitstream 3</Name>
[171]     <Version>0.1</Version>
[172]     </AssociatedTo>
[173]     <Type>Product Specification</Type>
[174]     <SpecificationRule>
[175]     <Description>If Service Template is '10Mbps $190', Bandwidth will be
10Mbps and Price will be $190.00.</Description>
[176]     <Type>ConditionalValue</Type>
[177]     <Target>
[178]     <Value>10</Value>
[179]     <SpecCharacteristicPath>SpecCharacteristics.SpecCharacteristic[2]</Spec
CharacteristicPath>
[180]     </Target>
[181]     <Target>
[182]     <Value>190.00</Value>
[183]     <SpecCharacteristicPath>SpecCharacteristics.SpecCharacteristic[3]</Spec
CharacteristicPath>
[184]     </Target>
[185]     <Condition>
[186]     <Parameter>
[187]     <SpecCharacteristicPath>SpecCharacteristics.SpecCharacteristic[0]</Spec
CharacteristicPath>
[188]     </Parameter>
[189]     <Parameter>Equals</Parameter>
[190]     <Parameter>10Mbps $190</Parameter>
[191]     </Condition>
[192]     </SpecificationRule>
[193]     </SpecificationRuleset>
[194]
```

[195] An example product specification instance can include XML data representing the following:

```
[196] ...
[197]     <ID>ORD000000887897</ID>
[198]     <AccessSeekerInteraction>
[199]     <ID>ORD000000887897</ID>
[200]     <billingAccountID>PTR000000000019</billingAccountID>
```

```
[201]   </AccessSeekerInteraction>
[202]   <description/>
[203]   <BusinessInteractionLocation/>
[204]   <BusinessInteractionRelationship/>
[205]   <Note>
[206]   <name>Order Note</name>
[207]   <description/>
[208]   <noteType>CustomerFacing</noteType>
[209]   <CreatorName>SEW</CreatorName>
[210]   <CreationDate>2013-12-09T00:43:48Z</CreationDate>
[211]   </Note>
[212]   <orderType>Connect</orderType>
[213]   <ProductOrderComprisedOf>
[214]   <quantity>
[215]   <amount>1</amount>
[216]   </quantity>
[217]   <action>ADD</action>
[218]   <AccessSeekerInteractionItem/>
[219]   <ItemInvolvesLocation>
[220]   <Place xsi:type="pla:TCFLocation">
[221]   <ID>LOC000001807002</ID>
[222]   <DescribedBy>
[223]   <value>WANGANUI</value>
[224]   <Characteristic>
[225]   <ID>Region</ID>
[226]   </Characteristic>
[227]   </DescribedBy>
[228]   <Geocode/>
[229]   <PhysicalAddress>
[230]   <StreetNumber>2</StreetNumber>
[231]   <StreetName>PORTAL</StreetName>
[232]   <StreetType>STREET</StreetType>
[233]   <Suburb>XXX</Suburb>
[234]   <City>YYYYY</City>
[235]   <postcode>4500</postcode>
[236]   </PhysicalAddress>
[237]   </Place>
[238]   </ItemInvolvesLocation>
[239]   <ItemInvolvesProduct>
[240]   <name>Bitstream 2</name>
[241]   <DescribedBy>
[242]   <value>Bitstream 2 100/50 10/2.5</value>
[243]   <Characteristic>
[244]   <ID>Service Template</ID>
[245]   <name>Service Template</name>
[246]   </Characteristic>
[247]   </DescribedBy>
```

```
[248]    <DescribedBy>
[249]    <value>58.75</value>
[250]    <Characteristic>
[251]    <ID>Price</ID>
[252]    <name>Price</name>
[253]    </Characteristic>
[254]    </DescribedBy>
[255]    <DescribedBy>
[256]    <value>None</value>
[257]    <Characteristic>
[258]    <ID>Enhanced Service Level</ID>
[259]    <name>Enhanced Service Level</name>
[260]    </Characteristic>
[261]    </DescribedBy>
[262]    <DescribedBy>
[263]    <value>None</value>
[264]    <Characteristic>
[265]    <ID>Diversity</ID>
[266]    <name>Diversity</name>
[267]    </Characteristic>
[268]    </DescribedBy>
[269]    <DescribedBy>
[270]    <value/>
[271]    <Characteristic>
[272]    <ID>E-NNI Port</ID>
[273]    <name>E-NNI Port</name>
[274]    </Characteristic>
[275]    </DescribedBy>
[276]    <DescribedBy>
[277]    <value/>
[278]    <Characteristic>
[279]    <ID>End User (Customer) Details</ID>
[280]    <name>End User (Customer) Details</name>
[281]    </Characteristic>
[282]    <CharacteristicValueReferences>
[283]    <value>John Smith</value>
[284]    <Characteristic>
[285]    <ID>Customer Name</ID>
[286]    <name>Customer Name</name>
[287]    </Characteristic>
[288]    </CharacteristicValueReferences>
[289]    <CharacteristicValueReferences>
[290]    <value/>
[291]    <Characteristic>
[292]    <ID>Business Name</ID>
[293]    <name>Business Name</name>
[294]    </Characteristic>
```

```
[295]   </CharacteristicValueReferences>
[296]   <CharacteristicValueReferences>
[297]   <value>041111332</value>
[298]   <Characteristic>
[299]   <ID>Contact Number</ID>
[300]   <name>Contact Number</name>
[301]   </Characteristic>
[302]   </CharacteristicValueReferences>
[303]   <CharacteristicValueReferences>
[304]   <value/>
[305]   <Characteristic>
[306]   <ID>Alternate Contact Number</ID>
[307]   <name>Alternate Contact Number</name>
[308]   </Characteristic>
[309]   </CharacteristicValueReferences>
[310]   <CharacteristicValueReferences>
[311]   <value>false</value>
[312]   <Characteristic>
[313]   <ID>Non Building Access Point</ID>
[314]   <name>Non Building Access Point</name>
[315]   </Characteristic>
[316]   </CharacteristicValueReferences>
[317]   <CharacteristicValueReferences>
[318]   <value>false</value>
[319]   <Characteristic>
[320]   <ID>Building Owner Consent Obtained</ID>
[321]   <name>Building Owner Consent Obtained</name>
[322]   </Characteristic>
[323]   </CharacteristicValueReferences>
[324]   <CharacteristicValueReferences>
[325]   <value>false</value>
[326]   <Characteristic>
[327]   <ID>End User Location is MUC</ID>
[328]   <name>End User Location is MUC</name>
[329]   </Characteristic>
[330]   </CharacteristicValueReferences>
[331]   </DescribedBy>
[332]   <DescribedBy>
[333]   <value/>
[334]   <Characteristic>
[335]   <ID>Technical Attributes</ID>
[336]   <name>Technical Attributes</name>
[337]   </Characteristic>
[338]   <CharacteristicValueReferences>
[339]   <value>Default</value>
[340]   <Characteristic>
[341]   <ID>DHCP Relay Agent</ID>
```

```
[342]          <name>DHCP Relay Agent</name>
[343]          </Characteristic>
[344]          </CharacteristicValueReferences>
[345]          <CharacteristicValueReferences>
[346]          <value>Default</value>
[347]          <Characteristic>
[348]          <ID>PPPoE Intermediate Agent</ID>
[349]          <name>PPPoE Intermediate Agent</name>
[350]          </Characteristic>
[351]          </CharacteristicValueReferences>
[352]          <CharacteristicValueReferences>
[353]          <value>Default</value>
[354]          <Characteristic>
[355]          <ID>UNI: Speed</ID>
[356]          <name>UNI: Speed</name>
[357]          </Characteristic>
[358]          </CharacteristicValueReferences>
[359]          <CharacteristicValueReferences>
[360]          <value>Default</value>
[361]          <Characteristic>
[362]          <ID>UNI: Duplex</ID>
[363]          <name>UNI: Duplex</name>
[364]          </Characteristic>
[365]          </CharacteristicValueReferences>
[366]          <CharacteristicValueReferences>
[367]          <value>Default</value>
[368]          <Characteristic>
[369]          <ID>UNI: Encapsulation</ID>
[370]          <name>UNI: Encapsulation</name>
[371]          </Characteristic>
[372]          </CharacteristicValueReferences>
[373]          </DescribedBy>
[374]          <version>0.1</version>
[375]          <SpecifiedBy xsi:type="prd:ProductSpecification">
[376]          <ID>SPC000000000004</ID>
[377]          <type>Product Specification</type>
[378]          <category>Product Specification</category>
[379]          </SpecifiedBy>
[380]          <validFor>
[381]          <startDateTime>2013-12-09T00:43:51.083106Z</startDateTime>
[382]          </validFor>
[383]          <ProductOffering>
[384]          <ID>POF000000000000</ID>
[385]          </ProductOffering>
[386]          </ItemInvolvesProduct>
[387]          </ProductOrderComprisedOf>
[388]          <AccessSeekerContact>
```

```
[389]          <contactName>XXX</contactName>
[390]          <emailAddress>XXX@gmail.com</emailAddress>
[391]          </AccessSeekerContact>
[392]          </ProductOrder>
[393]          </ManageProductOrderRequest>
[394]          ...
```

[395] Specifications and their associated instances may also be represented in JSON. Given JSON does not provide the same degree of syntactical regiment as XML some concessions need to be made (such as no direct support for XML Types). Therefore, when supporting multiple delivery channels with varying notation formats (ie. XML & JSON), XML constructs that are not easily represented in JSON may be avoided.

[396] The following example provides a summarised approach. In this structure the JSON name is the characteristic ID and JSON value is the instance value (for a given characteristic):

```
[397] {
[398]          "ItemInvolvesProduct":{
[399]          "name":"Bitstream 2",
[400]          "DescribedBy":[
[401]          {
[402]          "Service Template":"Bitstream 2 100/50 10/2.5",
[403]          "Price":"58.75",
[404]          "Enhanced Service Level":"None",
[405]          "Enhanced":"None",
[406]          "Diversity":"None",
[407]          "E-NNI Port":null,
[408]          "End User (Customer) Details":[
[409]          {
[410]          "Customer Name":"John Smith",
[411]          "Business Name":null,
[412]          "Contact Number":"0423444332",
[413]          "Alternate Contact Number":null,
[414]          "Non Building Access Point":false,
[415]          "Building Owner Consent Obtained":false,
[416]          "End User Location is MUC":false
[417]          }
[418]          ],
[419]          "Technical Attributes":[
[420]          {
[421]          "DHCP Relay Agent":"Default",
[422]          "PPPoE Intermediate Agent":"Default",
[423]          "UNI: Speed":"Default",
[424]          "UNI: Duplex":"Default",
[425]          "UNI: Encapsulation":"UNI: Encapsulation"
```

```
[426]          }
[427]          ]
[428]          }
[429]          ],
[430]          "version":"0.1",
[431]          "SpecifiedBy":{
[432]          "ID":"SPC000000000004",
[433]          "type":"Product Specification",
[434]          "category":"Product Specification"
[435]          }
[436]          }
[437]          }.
```

[438] An example order instance document can include the following:

```
[439] <BusinessInteraction i:type="CustomerOrder">
[440]       <InteractionStatus>
[441]          <ID>LCS000000000001</ID>
[442]       </InteractionStatus>
[443]       <InteractionDate>2015-06-01T03:31:01.405Z</InteractionDate>
[444]       <BusinessInteractionInvolves>
[445]          <PartyInteractionRole>
[446]             <InteractionRoleType>Submitter</InteractionRoleType>
[447]             <PartyRole>
[448]                <ID>PTR000000011068</ID>
[449]             </PartyRole>
[450]          </PartyInteractionRole>
[451]          <PartyInteractionRole>
[452]
<InteractionRoleType>CommerciallyResponsible</InteractionRoleType>
[453]             <PartyRole>
[454]                <ID>PTR000000020041</ID>
[455]             </PartyRole>
[456]          </PartyInteractionRole>
[457]       </BusinessInteractionInvolves>
[458]       <BusinessInteractionComprisedOf>
[459]          <BusinessInteractionItem i:type="CustomerOrderItem">
[460]             <Action>Add</Action>
[461]             <ItemInvolvesLocations>
[462]                <PlaceInteractionRole>
[463]                  <InteractionRoleType>ServiceAccessTo</InteractionRoleType>
[464]                    <Place>
[465]                       <ID>0010000000HA67C</ID>
[466]                    </Place>
[467]                </PlaceInteractionRole>
[468]             </ItemInvolvesLocations>
[469]             <ItemInvolvesInventoryEntities>
[470]                <InventoryEntity i:type="Product">
```

```
[471]                 <DescribedBy>
[472]                   <CharacteristicValue>
[473]                     <Value>John Doe</Value>
[474]                     <Characteristic>
[475]                       <ID> serviceProviderDetails applicantName </ID>
[476]                       <Name> Applicant Name </Name>
[477]                     </Characteristic>
[478]                     <SpecCharValueInstantiatedAs>
[479]                       <ValueType>String</ValueType>
[480]                     </SpecCharValueInstantiatedAs>
[481]                   </CharacteristicValue>
[482]                   <CharacteristicValue>
[483]                     <Value>0405783789</Value>
[484]                     <Characteristic>
[485]                       <ID> serviceProviderDetails contactNumber </ID>
[486]                       <Name> Contact Number </Name>
[487]                     </Characteristic>
[488]                     <SpecCharValueInstantiatedAs>
[489]                       <ValueType>DateTime</ValueType>
[490]                     </SpecCharValueInstantiatedAs>
[491]                   </CharacteristicValue>
[492]                 </DescribedBy>
[493]                 <SpecifiedBy>
[494]                   <ID>SPC000000008242</ID>
[495]                   <Type>Product</Type>
[496]                 </SpecifiedBy>
[497]                 <ManagedEntityRelationships/>
[498]                 <PartyInteractionRoles/>
[499]                 <PlaceInteractionRoles/>
[500]                 <OfferingDescribes>
[501]                   <ID>PRO000000005077</ID>
[502]                 </OfferingDescribes>
[503]                 <ExternalReferences/>
[504]               </InventoryEntity>
[505]             </ItemInvolvesInventoryEntities>
[506]           </BusinessInteractionItem>
[507]       </BusinessInteractionComprisedOf>
[508]       <DescribedBy/>
[509]       <SpecifiedBy>
[510]         <ID>SPC000000000002</ID>
[511]       </SpecifiedBy>
[512]       <Notes></Notes>
[513]       <ExternalReferences/>
[514]       <OrderCategory>Product</OrderCategory>
[515]       <OrderType>New</OrderType>
[516]       <CustomerReference>A2L</CustomerReference>
[517]     </BusinessInteraction>
```

[518] An example product specification document can include the following:

[519] <Specification xmlns="http://schemas.dgit.biz/telflow/cim/v3" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
[520] <ID>SPC000000000163</ID>
[521] <Name>Broadband New Connection</Name>
[522] <Description>Broadband New Connection<Description>
[523] <Version>2.0</Version>
[524] <DescribedBy />
[525] <Alias>NewConnectionProduct</Alias>
[526] <Type>Service</Type>
[527] <Status>Published</Status>
[528] <SpecCharacteristics>
[529] <SpecCharacteristic>
[530] <ID>serviceProviderDetails</ID>
[531] <Name>Service Provider Details</Name>
[532] <Description><Description>
[533] <Type>Collection</Type>
[534] <ReadOnly>false</ReadOnly>
[535] <MinCardinality>1</MinCardinality>
[536] <MaxCardinality>1</MaxCardinality>
[537] <SpecCharacteristicValues />
[538] <SpecCharacteristicRelationships>
[539] <SpecCharacteristicRelationship>
[540] <SpecCharacteristic>
[541] <ID>serviceProviderDetails applicantName</ID>
[542] <Name>Applicant Name</Name>
[543] <Description>we will set the applicantName value to a single value</Description>
[544] <Type>KeyValue</Type>
[545] <ReadOnly>false</ReadOnly>
[546] <MinCardinality>1</MinCardinality>
[547] <MaxCardinality>1</MaxCardinality>
[548] <SpecCharacteristicValues>
[549] <SpecCharacteristicValue>
[550] <Value />
[551] <ValueType>String</ValueType>
[552] <DefaultValue>Vodafone Wholesale</DefaultValue>
[553] <Default>false</Default>
[554] <SpecCharValueRelationships />
[555] </SpecCharacteristicValue>
[556] </SpecCharacteristicValues>
[557] <SpecCharacteristicRelationships />
[558] <DisplayFormat />
[559] </SpecCharacteristic>
[560] </SpecCharacteristicRelationship>
[561] </SpecCharacteristicRelationships>
[562] <DisplayFormat />

```
[563]       </SpecCharacteristic>
[564]       <SpecCharacteristic>
[565]         <ID>customerDetails</ID>
[566]         <Name>Customer Details</Name>
[567]         <Description></Description>
[568]         <Type>Collection</Type>
[569]         <ReadOnly>false</ReadOnly>
[570]         <MinCardinality>0</MinCardinality>
[571]         <MaxCardinality>1</MaxCardinality>
[572]         <SpecCharacteristicValues />
[573]         <SpecCharacteristicRelationships>
[574]           <SpecCharacteristicRelationship>
[575]             <SpecCharacteristic>
[576]               <ID>customerDetails firstName</ID>
[577]               <Name>First Name</Name>
[578]     <Type>KeyValue</Type>
[579]               <ReadOnly>false</ReadOnly>
[580]               <MinCardinality>0</MinCardinality>
[581]               <MaxCardinality>1</MaxCardinality>
[582]               <SpecCharacteristicValues>
[583]                 <SpecCharacteristicValue>
[584]                   <Value />
[585]                   <ValueType>String</ValueType>
[586]                   <Default>false</Default>
[587]                   <SpecCharValueRelationships />
[588]                 </SpecCharacteristicValue>
[589]               </SpecCharacteristicValues>
[590]               <SpecCharacteristicRelationships />
[591]               <DisplayFormat />
[592]             </SpecCharacteristic>
[593]           </SpecCharacteristicRelationship>
[594]         </SpecCharacteristicRelationships>
[595]         <DisplayFormat />
[596]       </SpecCharacteristic>
[597]     </SpecCharacteristics>
[598]     <SpecificationRelationships />
[599]     <SpecificationTags />
[600]     <PartyRoleCardinalities />
[601]     <PlaceCardinalities>
[602]       <PlaceCardinality>
[603]         <Type>ServiceAccessTo</Type>
[604]         <MinCardinality>1</MinCardinality>
[605]         <MaxCardinality>1</MaxCardinality>
[606]       </PlaceCardinality>
[607]     </PlaceCardinalities>
[608]   </Specification>
```

[609] An example lifecycle document can include the following:

```
[610]  <Lifecycle>
[611]        <ID>LCX000000000001</ID>
[612]        <Name>Telflow Default</Name>
[613] Version>1.0</Version>
[614]        <Status>Active</Status>
[615]        <LifecycleStates>
[616]              <LifecycleState>
[617]                    <ID>LCS000000000001</ID>
[618]                    <Name>Acknowledged</Name>
[619]                    <Type>Start</Type>
[620] </LifecycleState>
[621]              <LifecycleState>
[622]                    <ID>LCS000000000002</ID>
[623]                    <Name>In Progress</Name>
[624] <Type>Intermediate</Type>
[625]              </LifecycleState>
[626]              <LifecycleState>
[627]                    <ID>LCS000000000003</ID>
[628]                    <Name>Complete</Name>
[629] <Type>EndSuccess</Type>
[630]              </LifecycleState>
[631]              <LifecycleState>
[632]                    <ID>LCS000000000004</ID>
[633] <Name>Cancelled</Name>
[634] <Type>EndFailure</Type>
[635]              </LifecycleState>
[636]        </LifecycleStates>
[637]        <LifecycleTransitions>
[638]              <LifecycleTransition>
[639]                    <Name>Submit</Name>
[640]                    <Output>
[641]                          <ID>LCS000000000001</ID>
[642]                          <Name>Acknowledged</Name>
[643]                          <Type>Start</Type>
[644]                    </Output>
[645]                    <PartyRoleTypes>
[646]                          <PartyRoleType>Customer</PartyRoleType>
[647]                          <PartyRoleType>Provider</PartyRoleType>
[648]                          <PartyRoleType>System</PartyRoleType>
[649]                    </PartyRoleTypes>
[650]              </LifecycleTransition>
[651]              <LifecycleTransition>
[652]                    <Name>Accept</Name>
[653]                    <Input>
[654]                          <ID>LCS000000000001</ID>
[655]                          <Name>Acknowledged</Name>
```

```
[656]                       <Type>Start</Type>
[657]                   </Input>
[658]                   <Output>
[659]                       <ID>LCS000000000002</ID>
[660]                       <Name>In Progress</Name>
[661]                       <Type>Intermediate</Type>
[662]                   </Output>
[663]                   <PartyRoleTypes>
[664]                       <PartyRoleType>Provider</PartyRoleType>
[665]                       <PartyRoleType>System</PartyRoleType>
[666]                   </PartyRoleTypes>
[667]               </LifecycleTransition>
[668]               <LifecycleTransition>
[669]                   <Name>Complete</Name>
[670]                   <Input>
[671]                       <ID>LCS000000000002</ID>
[672]                       <Name>In Progress</Name>
[673]                       <Type>Intermediate</Type>
[674]                   </Input>
[675]                   <Output>
[676]                       <ID>LCS000000000003</ID>
[677]                       <Name>Complete</Name>
[678]                       <Type>EndSuccess</Type>
[679]                   </Output>
[680]                   <PartyRoleTypes>
[681]                       <PartyRoleType>Provider</PartyRoleType>
[682]                       <PartyRoleType>System</PartyRoleType>
[683]                   </PartyRoleTypes>
[684]               </LifecycleTransition>
[685]               <LifecycleTransition>
[686]                   <Name>Cancel</Name>
[687]                   <Input>
[688]                       <ID>LCS000000000001</ID>
[689]                       <Name>Acknowledged</Name>
[690]                       <Type>Start</Type>
[691]                   </Input>
[692]                   <Output>
[693]                       <ID>LCS000000000004</ID>
[694]                       <Name>Cancelled</Name>
[695]                       <Type>EndFailure</Type>
[696]                   </Output>
[697]                   <PartyRoleTypes>
[698]                       <PartyRoleType>Customer</PartyRoleType>
[699]                       <PartyRoleType>Provider</PartyRoleType>
[700]                       <PartyRoleType>System</PartyRoleType>
[701]                   </PartyRoleTypes>
[702]               </LifecycleTransition>
```

[703] <LifecycleTransitions>
[704] </Lifecycle>

Interpretation

[705] Many modifications will be apparent to those skilled in the art without departing from the scope of the present invention.

[706a] Throughout this specification, unless the context requires otherwise, the word "comprise", and variations such as "comprises" and "comprising", will be understood to imply the inclusion of a stated integer or step or group of integers or steps but not the exclusion of any other integer or step or group of integers or steps.

[706] The reference in this specification to any prior publication (or information derived from it), or to any matter which is known, is not, and should not be taken as an acknowledgment or admission or any form of suggestion that the prior publication (or information derived from it) or known matter forms part of the common general knowledge in the field of endeavour to which this specification relates.

THE CLAIMS:

1. A computer-implemented process for defining a telecommunications product in accordance with a predefined data schema, the process including the steps of:

receiving, via a user interface, a request from a user for a specification for the telecommunications product;

receiving, via the user interface, characteristics data representing one or more characteristics for the specification from the user, including a plurality of properties for at least one of the characteristics;

receiving, via the user interface, input rules data representing at least one rule for the specification, including:

a selection of at least one of the received characteristics to provide a condition characteristic for the rule to operate,

a selection of a rule type to define operation of the rule, and

a selection of at least one of the received characteristics to provide a target characteristic on which the rule will operate;

processing the characteristics data and the input rules data, according to the predefined data schema, to generate specification data representing the characteristics in a specification document including a plurality of fillable data fields, and output rules data, in a rules document representing the rules for filling the fillable data fields,

such that the specification document and the rules document are useable together to:

generate order interface data, representing an order interface for a customer to request the product, including one or more of the properties by filling data into fillable data fields,

generate instance data representing an instance document including the data filled into the fillable data fields based on user input from the customer and data generated by processing the rules document in a rules engine, wherein the instance data, when re-combined with the specification document, represent product components of the requested product for provisioning the product.

2. The process of claim 1, including the step of using the specification data to render an order interface for the product.

3. The process of claim 1 or 2, including the steps of:

receiving a selection of at least one relationship for one of the characteristics, wherein the relationship specifies a different previously defined product specification;

displaying the relationship; and

generating relationships data defining the relationship.

4. The process of claims 1 – 3, including the step of validating the instance data representing an instance of an order, generated using the order interface, against the specification data and the rules data.

5. The process of claims 1 – 4, including the step of decomposing an instance of an order, generated based on the specification data, the rules data and relationships data, to generate decomposed order data representing a plurality of telecommunications services which may be provided by a supplier according to known standards.

6. A process for provisioning a telecommunications product, the process including the steps of:

the process of claims 1 – 5;

generating the order interface data; and

generating the product component data representing the product components.

7. The process of any one of claims 1 - 6, including:

storing the specification data separately from the rules data.

8. The process of any one of claims 1 - 7, including generating the specification document and the rules document using the predefined data schema such that the specification document and the rules document comply with the predefined data schema.

9. The process of claim 8, including validating the generated specification document, the generated rules document, and the generated instance document with the predefined data schema.

10. The process of any one of claims 1 - 9, including storing the instance data separately from the associated specification data.

11. The process of claim 10, including using a rendering engine to display the instance data in context using the associated specification data.

12. The process of any one of claims 1 - 11, including updating status data representing a lifecycle status of an instance document based on user input, wherein the lifecycle status is selected from a plurality of predefined statuses in a lifecycle document associated with the associated specification.

13. A system, including:

a composer module that:

receives, via a user interface, a request from a user for a specification for the telecommunications product,

receives, via the user interface, characteristics data representing one or more characteristics for the specification from the user, including a plurality of properties for at least one of the characteristics,

receives, via the user interface, input rules data representing at least one rule for the specification, including:

a selection of at least one of the received characteristics to provide a condition characteristic for the rule to operate;

a selection of a rule type to define operation of the rule; and

a selection of at least one of the received characteristics to provide a target characteristic on which the rule will operate, and

processes the characteristics data and the input rules data, according to the predefined data schema, to generate specification data representing the characteristics in a specification document including a plurality of fillable data fields, and output rules data, in a rules document representing the rules for filling the fillable data fields,

such that the specification document and the rules document are useable together to:

generate order interface data, representing an order interface for a customer to request the product, including one or more of the properties by filling data into the fillable data fields, and

a rendering engine, including a rules engine, that generates instance data representing an instance document including data filled into the fillable data fields based on user input from the customer and data generated by processing the rules document in the rules engine, wherein the instance data, when re-combined with the specification document, represent product components of the requested product for provisioning the product.

14. The system of claim 13, including a specification module that stores the specification data separately from the rules data.

15. Non-transitory storage with computer-readable instructions configured to implement the process of any one of claims 1 – 12.