2.3 Product Requirements

The product requirements model specifies the behavior that a product is expected to exhibit, within a specified operational context[1]. This model is intended to establish a shared developer understanding of a problem-solution that is a consistent elaboration of customer needs as specified in the product delivery model.

The product's behavior is defined in terms of interactions with entities specified in the product environment model as constituting its operational environment. This model defines the "competence" of the product in exhibiting behavior: experience in this case is how envisioned behavior has previously been realized within the enterprise; expertise is its expected ability to enact reasoned/planned action; and knowledge is its awareness of the composition and functioning of the supported customer enterprise and ecosystem.

During development, the requirements model is an evolving "build-to" specification of the product, defining all aspects of the observable behavior that the product is being built to exhibit. Upon deployment of the built product into operational use, the requirements model defines the correct "as-built" behavior of the product. The evolution of a product is viewed in terms of the aspects of its observable behavior that are changing.

Requirements Model Elements

The product environment model identifies the product environment entities that the product can reference both as sources of information and as agents for initiating actions that actualize the product's behavior.

Whereas customer needs specifies the capabilities the customer expects the product to support, the product requirements model prescribes the product's expected observable behavior from a developer perspective; this describes the product from 5 perspectives:

---

[1] Loosely derivative of: K.L.Heninger, "Specifying Software Requirements for Complex Systems: New Techniques and Their Application", IEEE Trans on Sw Eng SE-6(1), Jan 1980.

- Intention – a concise description of the product's purpose, in providing capabilities that support customer needs and objectives
- Information – a model of the product's information space on which behavior is based and how its content is determined
- Behavior – a specification for the product's interactions with entities in its environment to induce effects on the ecosystem
- Quality – a characterization of the degree to which the product is expected to exhibit specified behavioral qualities
- Constraints – a characterization of extrinsic factors that limit the product's realization

Product Intention

The product intention element specifies the development project's understanding of why the product is needed and what benefit it offers to the customer in its operational context. This element is the product "vision" from a developer perspective. It concisely defines a needs-based rationale for the product. Its purpose is to establish a shared developer understanding of the product's expected capabilities in terms of its role in customer operations. This should suffice in describing any realizable product that conforms to customer current and likely future needs. This element is augmented with definitions of key terminology used to express the product intention and references to materials that further explain the nature of the problem-solution space that it addresses.

Product Information

The product information element specifies the abstract information space within which the product operates. This space represents all information about the ecosystem (environment and associated entities) that is relevant to product operation. This element represents the changing state (past, present, and future, as appropriate) of the ecosystem and any inherent dynamics that can cause the state to change. This model defines the concepts, attributes, and relationships upon which reasoning is based and associated computational logic that specifies how corresponding information content

changes. Content is obtained from entities operating in the product's environment or derived based on information dependencies that describe how content changes.

Product Behavior

The product behavior element specifies observable effects on its operational environment that the product can initiate using the services of associated logical entities. Entities provide capabilities that the product can engage to create observable behavior. Each effect is specified as a function that defines the value of an output to an entity as a computation on information model content.

Product Quality

The product quality element specifies the criteria by which the product is built and evaluated as acceptable for its intended use. This element is a specification of the product quality factors that any product realization must satisfy to be acceptable. Product quality is the degree to which a product provides capabilities that will satisfy customer needs.

Product quality is an aggregate measure of quality factors organized into four broad categories; each of these categories encompasses a flexible set of finer-grained properties that each to varying degrees influence the fit and acceptability of a product for its intended purpose. The product design will determine the significance of the various factors in the product realization. The relative importance and sensitivity of the various factors provide a basis for tradeoffs among alternative solutions.

Product Constraints

The product constraints element specifies extrinsic, externally-imposed factors that limit options for the product's realization. These factors can reflect provider- or customer-established conventions concerning their operations, industry conventions that constitute standard practices, or government-imposed criteria for compliance with legal, environmental, safety, or security standards.

Potential constraints include: aspects of the operational (deployment) platform, enterprise policies and practices, regulations:legal/contractual, and customer

conventions or preferences. Extrinsic constraints that cannot be relaxed may preclude otherwise potentially preferable alternative solutions.

Product Evolution

The requirements model is developed incrementally, in keeping with the product master plan and associated increment plans, to specify a series of product releases. These plans can change as customer needs and circumstances change. Each increment plan defines how the product is to be modified or enhanced, toward meeting customer needs for a subsequent release. The requirements model is routinely modified in each increment as guided by these plans.

At the same time, changes in developer understanding of customer needs and how the product can be built to satisfy those needs lead to changes in the requirements model. The requirements model also changes as a result of changes in other elements of the project model:

- Changes in acceptance criteria as specified in the customer relationship element of the project management model;
- Changes in the customer needs element of the product delivery model corresponding to changes in actual needs;
- Changes in the product master plan or increment plan elements of the project management model due to unexpected progress or impediments (e.g., feasibility, cost, or resources) resulting in changes in product capabilities to be built;
- Changes in other elements of the product model, based on improved developer understanding of the problem or solution feasibility, that result in inconsistency with the requirements model.

Areas of Competence on Which Requirements is Based

For a proper understanding of customer needs, the developer needs access to persons knowledgeable in how the customer enterprise operates. Customer agents having competence and authority to accurately convey the needs of the targeted customer enterprise should be active participants in the development. Developers, assisted by

program marketing and customer relations along with assigned customer agents as appropriate, must have appropriate competence in the area.

The product requirements model is developed based on two primary sources of competence:

- Business area competence – the *knowledge* of the developer in the theory and practice of the type of enterprise targeted by the product; familiarity with *experiences* with similar and related systems and in developing similar and related products; *expertise* that the developer has with relevant techniques and technology appropriate for developing products of the sort needed
- Customer needs – a customer perspective (conveyed in the customer needs element of the product delivery model) on how the problem-solution is perceived in terms of the purposes that a product will support, the capabilities that the product will provide in support of their operations, and any constraints that an acceptable solution must satisfy

Customer needs, being an expression of customer criteria for product acceptance, is an under-constrained specification of the envisioned product in that many different products could be realized that would satisfy such criteria. To the degree that customer needs under-specify a product, the developer has flexibility to make engineering tradeoffs for a viable best-fit solution.

By applying business area competence to customer needs, the product requirements specifies the observable behavior of a specific product to be realized. The result is an over-constrained specification of the envisioned product, reflecting developer judgement in identifying, exploring, and resolving uncertainties, alternatives, and tradeoffs while elaborating and refining developer understanding of customer needs. This has the effect of reducing a candidate set of potentially acceptable products to a single product to be realized.

In developing a product through a series of increments, each increment targets reduced capabilities relative to specified customer needs. Subsequent increments add or revise capabilities over time to more closely match understood needs. At the same time, actual

or understood customer needs may change, possibly differing from prior increments. Increments continue until a version is completed that is acceptable for delivery. In exploring alternatives, the developer may identify changes in customer needs that will make better or less costly solutions feasible, leading in appropriate cases to a collaborative revision of customer needs, with project and customer management involvement if terms of the specified customer relationship inhibit those changes. A delivered version may still lack capabilities that have been deferred, with customer concurrence, to a future release reflecting cost-schedule constraints.

<———————

Building the right product requires an understanding of the customer enterprise, how that enterprise operates, how the envisioned product will affect that operation, and relevant subject matter. Such understanding comes most easily from having worked in such an enterprise or having built products that solved similar problems, recognizing how differences in customers' circumstances can affect a solution. This understanding may be supplemented through customer-provided information, or from program marketing as a customer proxy. Needs that are initially incomplete, not well understood, or poorly communicated will require refinement through customer-provider collaboration as the product is developed.

Customer needs that are over-constraining, exhibiting uncertainties, ambiguities, or nonessential constraints, can impose excessive solution complexity, portending delay, increased cost, or infeasibility. The developer may propose revisions that will mitigate or defer such complexity, allowing the customer to influence the tradeoff between needs and effort. An alternative is for the developer to create an interim more limited version of the product that can later be revised to properly address the need. A sound practice is to first build a product version that satisfies a subset of expressed needs, with subsequent efforts to develop more complete versions.

———————->

Specifying Model Elements

{provide details about forms of specifying "behavior"; include modes/user-roles, conditional invocation-/event-/demand-driven behavior upon each entity, ?; the form in which model elements are specified can differ depending on the nature of the product being developed; these characterizations will give only a general sense of the elements of form that such specifications should exhibit}

Specifying Product Intention

The product intention element specifies in concise terms for a shared realization-independent understanding by developers the purpose of the product to the customer. This includes the nature of the operation in which it is used, the type of information it manages, and the type of capabilities that it provides.

Specifying Product Information

{specification of the product information space as an active semantic data model as mapping from the product environment model}

Specifying Product Behavior

Behavior is expressed in a formalism that is characteristic of the sorts of interactions that a given type of entity accommodates (i.e., the actions that it recognizes as defined in the product environment model). Each such formalism has two elements: the criteria by which the action is initiated and how the content of the action is determined.

The product environment model specifies the entities that are accessible to the product and the set of addressable actions associated with each entity. An action is the dispatch of a request and associated data. Each action is associated with a particular type of device (edge, user role, or system) whose specification determines the form that valid actions can take.

{This will be described more clearly once the content of the product environment model is more complete. Defining behavior this way is easy to enough (given that the intended behavior is understood) but need to avoid getting overly specific here to particular formalisms.}

Action Criteria

Action criteria is a specification of the operational condition that causes the action to be initiated. Actions can be initiated based on one or any combination of (1) a specific request (i.e., event) initiated by another action, (2) a transition in the value of a predicate concerning the content of monitored data, or (3) a transition in operational mode of the product instance. When given criteria is satisfied, the associated action is initiated according to its specified effects as appropriate to the type of device it targets.

Edge Device Actions

An edge device initiates actions that may affect the environment as needed to realize intended goals of the product's operation.

Interface Device Actions

An interface device initiates actions for coordinating actions or sharing information with other entities operating in the same operational environment. Each such action is defined in a form that is known and can be acted on designated entities.

User Role Device Actions

A user role device is a virtual interface device specifically concerned with interactions via one or more logical interface devices with operators and users of product capabilities. A user role device is characterized as supporting a specific user/operator "role". Each role is specified as allowing a coherent set of responsibilities in enterprise operations. A user role device presents data to convey information within the purview and operational context of users performing the activities associated with a designated role. Data may have associated mechanisms associated with a given type of user interface device for responding to user actions.

System Device Actions

A system device transfers action requests or data to one or more specified interface devices defined as associated with a specified type of system entity. Each interface

device is defined in terms of the form in which understood actions and data can be accepted.

Specifying Product Quality

In the context of software-based products, product quality is viewed as "behavioral" quality, that being the degree to which the product addresses all aspects of quality to be exhibited in its observable behavior. This quality can be expressed in four facets: functionality, performance, dependability, and usability. While all four categories affect every product, the specific properties encompassed by each and the degree of influence that each property has on acceptability of the product can differ based on the nature of the product and circumstances of its intended use. The requirements model defines which specific properties apply to the product and how much flexibility there is in each property to enable tradeoffs among properties to resolve conflicts among them. (A notional formulation of the constituent elements of behavioral quality is provided below.)

{explain how quality factor goals are expressed as objective that guides design-implementation analyses of alternative & tradeoffs and product evaluation, either as a "lower-bound" or value range.}

{explain how quality factor goals may differ across different parts of a product (e.g., a factor such as safety may need to be directly addressed within only a portion of the product architecture)}

Notional Behavioral Qualities (1)

Functionality, the degree to which a product exhibits expected behavior (i.e., effectiveness); typical elements include:

- *Utility* (satisfies its intended purpose including aligning with and supporting encompassing enterprise operations)
- *Interoperability* (operates and communicates properly with entries in its ecosystem)
- *Adaptivity* (has the means to modify its own behavior as directed (i.e., reconfigurability for e.g., localization, specialization, or personalization) or in response to prescribed circumstances such as high demand, fault, or degraded conditions)

Performance, the degree to which a product supports an anticipated workload, consistent with available resource capacities (i.e., efficiency); typical elements include:

- *Responsiveness* (reacts to external stimuli and internal effects sufficiently to maintain consistency internally and with respect to its ecosystem)
- *Utilization* (makes best use of available resources, including energy and environmental efficiencies achievable within equipment capabilities)
- *Conservance* (minimizes use of physical assets and logistical resources and impact on environmental resources and conditions)
- *Throughput* (produces effects at a rate sufficient to maintain consistency with external elements of the environment and ecosystem)
- *Scalability* (adjusts to projected variations in workload and resource capacities)

Notional Behavioral Qualities (2)

Dependability, the degree to which a product continues to produce expected effects (behavior and data) under all (normal, abnormal, and unforeseen) conditions; typical elements include:

- *Reproducibility* (exhibits differences in behavior or information content only as a result of differences in external influences)
- *Reliability* (exhibits consistent and accurate effects under normal and degraded conditions) {includes predictability, stability, and survivability, including resilience and recoverability}
- *Availability* (operates without interruption or unavailability of information) {includes continuity and connectivity}
- *Integrity* (provides acceptably accurate and complete information, while excluding unauthorized access and precluding any unspecified behavior) {includes fidelity and privacy/security}
- *Safety* (prevents or detects and mitigates conditions or actions that would cause unintended damage to any part of itself or its ecosystem) {includes avoidance of false, misrepresented, distorted, unethical, or biased content}

Usability, the degree to which a product is able to be used properly by its intended users/operators; typical elements include:

- *Conformability* (operates consistent with legal, financial, ethical, and equity dictates and the competencies (language-terminology, knowledge-expertise, and skills-abilities) of users) {includes accessibility}
- *Learnability* (aids and instructs users in the nature and use of its capabilities and information content)
- *Explainability* (is able to communicate causes and rationale for prior or prospective behavior and information content) {includes transparency and accountability}
- *Aesthetics* (presents and properly conveys provided information and functionality in a form that facilitates intended usage) {includes consistency}

Specifying Product Constraints

The product as-built may be constrained to conform to external conventions, customer criteria, and program conventions. Such constraints may limit engineering alternatives that would otherwise be left to developer judgement based on identification and analyses of feasible alternatives. The product must be built in anticipation of any changes in these aspects to suit current and future customer circumstances.

External constraints include governmental and industry conventions. Customer-imposed constraints relate to how the product is expected to fit within and support customer operations. Program constraints relate to conventions for interaction among products and with entities specified in the product environment model.

External Constraints

Constraints may be imposed on products by external authority. Formal authority includes the various legal and regulatory jurisdictions that establish standards for accountability, infrastructure, communication, data privacy, and environmental impacts. Relevant industry and market governing bodies may establish additional technical conventions. The product must be built to conform to the constraints associated with its intended operational environment(s).

Customer-imposed Constraints

The project management customer relationship element and the customer needs specification of the product delivery model may identify (or imply) constraints arising from customer enterprise or operational conventions to which the product will need to conform.

These constraints may designate interface conventions associated with tools and technologies or systems that are related in some way with the enterprise's envisioned use of the product. These may entail expected use of the customer's computational environment, particular commercial tools or devices, data security protocols, monitoring and auditing practices, or other organizational conventions that the product must be built to satisfy.

Customer policies reflect consideration of how the product will fit into the customer environment and operations (e.g., expected use of existing or planned facilities, use of previously established or new practices). These policies affect the platform on which the product is built to operate and possibly the practices used in building it. Prescribed policies may constrain the resolution of development alternatives.

Examples of policies that could be imposed include:

- Designated use of preferred commercially-available ("COTS") products (tools, data storage, computational platform)
- Computational platform standards and conventions regarding computation and data distribution, resource usage, messaging, transaction protocols, and logging practices
- Platform management protocols (startup/shutdown, hardware health and diagnostic conventions, concurrency techniques, fault handling and degraded processing)
- Platform integrity (system, transactional, and data access security and safety practices)
- Hardware-software configuration, compatibility, and reconfiguration/reprogramming practices
- Deployment practices (installation, training, logistics)
- Accommodations for future operational and computational environment evolution

Program-based Constraints

A program, consistent with enterprise guidance, may establish conventions for how its products are to interact with instances of related products and with relevant entities in its operational environment. Interactions may be constrained as to the entities and products to be referenced and the protocols to be used in communicating with these. (These are in addition to developmental practices established by project management.)