1.2 Basic Terminology

The proper understanding of this text depends on understanding the terminology being used. This is a particular challenge in the context of software methodology where commonly used terms are often not commonly, or even clearly, defined or understood.

Some definitions given here may differ from conventional usage or a reader's prior understanding but these differences are not arbitrary. They differ for any of several reasons: to expand prior unnecessarily parochial definitions of a term; to resolve ambiguities in the usage of terms with similar meanings, sometimes used interchangeably; to align more closely with how a term is defined in other fields; or simply to better convey meaning within the scope of the presented methodology. An appreciation of such differences will in fact provide some initial insight into how the approach described here differs from other approaches.

Terminology, Type 1

Processes, Methods, and Methodology

A ***process*** is a set of delineated activities that are performed as needed to achieve an associated objective. An ***activity*** is characterized by its purpose and the knowledge and expertise it requires to perform. A ***method*** is guidance and criteria that prescribe a systematic, repeatable technique for the performance of a given activity. A ***task*** is an assignment of resources for the performance within a specified scope of one or more related activities following a specified method or practices. A ***methodology*** is an integrated body of principles, practices, and methods that prescribe the nature and proper performance of the whole of a specified process. A given methodology may accommodate differing approaches by allowing for alternative practices and methods, as long as conformance to its defining principles is maintained. This text describes a methodology for the engineering and manufacture of software-based products.

Objectives and Goals

Due to ambiguity in common usage of "objective" and "goal", these are defined in terms of how they are to be understood herein. An ***objective*** denotes the motivation for some undertaking, an overarching but informally-defined aspiration for what is to be

accomplished. An objective has an associated set of ***goals,*** each identifying (consistent with quality improvement approaches) specific success criteria. Success criteria is defined in terms of appropriate metrics for measuring (objectively or subjectively) the degree to which each particular goal has been satisfied. Goals provide a concrete basis for evaluating the progress an endeavor has made toward achieving its objectives.

Models and Documents

A ***model*** is a (simplified or partial) representation of an item, sufficient as a basis for obtaining approximate answers to a designated set of questions about that item. A model provides indicated information about an associated item without requiring the item itself to be inspected or even accessible. The purpose of a model is determined by the questions it can be used to answer. Answers must be accurate relative to the referenced item but may be only "approximate" in that more exact answers would require an unwarranted costly or otherwise infeasible direct inspection of the item itself.

A ***document*** (or "documentation") is an identifiable organized communication of information on some topic for a designated audience. A document may be a primary source, serving as a (typically informal) model of an item, a supporting source of explanatory material for another item, or a secondary source providing content derived from models or documentation concerning one or more other items.

Specifications and Realizations

A dependency between two items is represented as a "specification-realization" pair. A ***specification*** is a model that prescribes the criteria that another item of interest must satisfy to be considered valid. A ***realization*** is a model that is required to satisfy an indicated specification. Consistency between a specification and its realization must be positively confirmed. Any change in either part of a previously consistent specialization-realization pair establishes an inconsistency that may require further changes in either or both parts of the pair to reestablish consistency.

Terminology, Type 2

Enterprises, Programs, and Projects

An ***enterprise*** is a managed undertaking established by one or more chartering entities (including persons, legal entities, and governmental entities) to pursue an identified purpose. An enterprise is managed by means of processes or practices that it institutes to pursue its purpose and operates by its chartering and direction of programs that will contribute toward its purpose.

A ***program*** is a managed undertaking of an enterprise intended to operate within a defined focus and scope corresponding to a specified area of institutional competence. A ***project*** is a managed effort chartered by a program to undertake actions within an allocation of its scope, objectives, and resources. A program may be the governing authority over a set of related projects.

Customers and Markets

A (potential) ***customer*** is an enterprise (person or organization) having an endeavor that could be improved in efficiency and effectiveness through the deployment and use of an appropriate product (i.e., one having capabilities conducive to such endeavors).

A ***market*** is a set of potential customers for a product having an associated purpose. A market is "coherent" if encompassed customers have similar needs, thus being amenable to accepting similar products, according to their specific needs and practices.

A "simple" market is a coherent market that consists of a single customer or in which all customers are accepting of the same product, willing and able to adjust their operations as required to fit the product, rather than requiring a product that has been designed to fit their specific needs and preferred practices. By convention, the terms "customer" and "simple market" will be considered equivalent and may be used interchangeably.

Providers and Acquirers

A ***provider*** is a program that charters projects for the development of products that support the needs of a specified, internal or external, market. A provider may also be a customer to providers of products that support its own efforts.

Broadly, an ***acquirer*** is a potential customer in a provider's designated market, or in a narrower sense, a project chartered by such a customer to serve as its proxy, authorized to represent customer needs and interests to providers. Ideally, an acquirer project includes staff who are knowledgeable and experienced in the customer's business, technology, and operations. An acquirer must ensure that customer views are accurately represented to providers in that providers will view the acquirer as speaking authoritatively for the represented customer.

In targeting a simple market consisting of multiple customers, a provider project may need to rely on a more indirect, less definitive determination of their customers' needs. This entails interacting with proxies who understand the endeavors of the envisioned customer community and can represent their needs to a provider. Such proxies may include experienced individual users, qualified subject matter experts, past developers of similar products, and provider marketing or sales personnel in addition to representatives of selected customers.

Terminology, Type 3

Environments, Entities, and Systems

An ***environment*** is a space-time (i.e., reality) within which entities of interest operate. An ***entity*** is a person or device, or coordinated aggregate thereof, having agency (being able to undertake certain actions). An environment or entity can be physical, virtual, or augmented (a physical-virtual hybrid) and is characterized by properties that manifest the effects of associated naturally-occurring and entity-initiated phenomena.

A ***system*** is an aggregate entity, representing the collective behavior of an associated set of entities. The specific purpose and extent of a system, and its composition, is delineated subjectively based on which entities are perceived as being agents of that system. The separate behaviors and interactions of its associated entities are attributed to the system as a whole, determining how it is perceived as directly affecting both the containing environment and other entities sharing that environment.

An ***ecosystem*** is an environment and the entities that operate within it. A system can be influenced by the behavior of other entities operating in the same environment, either

indirectly via an entity's effects upon the properties of the shared environment or directly via an entity having a means to share information or coordinate action. Entities operating within a shared environment may be seen to be interacting cooperatively, competitively, or equitably with regard to their respective purposes.

People participate in an ecosystem (1) through direct interactions with other people, (2) as ***users*** of devices that provide access to the mechanisms and information content of a system, and (3) as ***operators*** of equipment to monitor and control associated detection and initiation of physical phenomena. Relative to a given system, a person is characterized by the ***roles*** that they are authorized to perform within that system. A role corresponds to rights a person has to access information and capabilities of a product according to their responsibilities in the operation of an enterprise.

Information, Data, and MetaData

Reality is the properties and phenomena that characterize the evolving space-time of a physical, virtual, or hybrid realization of an ecosystem. ***Information*** is a selective characterization of reality from the perspective of one or more observers (i.e., entities). Specific aspects of reality can be static, intermittently changing, or continuously changing over time. The information associated with an environment or entity is defined abstractly as its "information space", only some elements of which may be susceptible to being directly observed or modified. The purpose of a system is to perform actions that will obtain or change information associated with relevant aspects of reality.

Data are a representation of the accessible information that a system needs to perform its intended purpose. An entity associated with a system may obtain data directly by measurements of properties or phenomena or indirectly by derivations based on other logically related data (including prior, historical, or theoretically determined values). Data to be useful must be sufficiently precise, accurate, and timely in representing corresponding information. However, the validity of data vis-a-vis information may be limited by the mechanisms available to observe or derive the properties and phenomena exhibited within an ecosystem. The value of data associated with undetermined information is "undefined".

Metadata indicate how data relate to information (as an approximation of reality). Metadata can be used to determine and maintain the basis of data values and initiate actions to update those values as associated information changes. At the least, metadata details the identity, provenance (i.e., when and how a value has been determined), and representation (i.e., how information is encoded as data) of associated data. Metadata can also provide a basis for evaluating confidence in the validity of data, including degree of certainty that data values have been correctly determined, that values of related data are consistent, and that data is a sufficiently close approximation to associated information for its intended use. Using metadata to discover and correct or mitigate discrepancies in data vis-a-vis information can be an important aspect of a system's operation.

Hardware, Software, and Platforms

Hardware is any physical apparatus (e.g., biologic, chemical, electronic, mechanical), or appropriate aggregation thereof, whose purpose is to actualize a conceived process. ***Software*** is an encoding by which such a process can be expressed, broadly in terms of obtaining, transforming, deriving, storing, and dispensing information encoded in an analog or digital form as data. Software operates through the medium of hardware built or determined to be suitable for enacting its intended behavior.[1]

In general, software is a specification of behavior that is realized in a form that is expected to induce that behavior in associated hardware. More specifically, software is a human-decipherable ("source") representation of intended behavior that has an equivalent ("object") representation that specifies how it will cause compatible hardware to behave.

From a software perspective, it is useful to distinguish between two sorts of hardware: platforms and devices. A hardware platform is the physical infrastructure (i.e., physical housing and power, communications, and environmental hardware) that enables a collection of devices to operate as an integrated unit (e.g., a computer, a cellphone, a

[1] An electronic apparatus directly enacts software-encoded behavior. In other cases, the mechanism may differ: a chemical or mechanical apparatus can be constructed to produce behavior corresponding to a software encoding of its intended operation; similarly, viewing DNA as a type of software encoding, genomic machinery enables an organism to enact DNA-specified behaviors.

satellite, an automobile, a factory, a communications network), coordinating activity and sharing data.

A device (sometimes referred to as an "instrument" or in aggregate as "equipment" or "machinery") provides specific capabilities for detecting or producing physical phenomena in the ecosystem within which it operates. A set of physically distinct devices can be interconnected by a platform to provide an aggregate capability, in effect creating a composite device. The behavior of each device is determined by its implementation (e.g., the fidelity with which it detects reality), limited by the capabilities of the platform on which it operates (e.g., speed and latency of communicating between two devices).

A device will have any of three types of supporting functionality: computation, data storage, and portage. The nature of a given device is determined by the relative significance of these to its essential purpose and use (including ancillary functionality it needs to support its own operation). A *computational device* is hardware whose primary purpose is to effect software-defined behavior. A *data storage device* is hardware whose primary purpose is to persistently store data representing information over time about the environment and entities with which computation is concerned. A portage device receives, stores, analyzes/filters/converts, and transmits data that corresponds to information associated with its containing environment or associated entities. An *edge device* is portage hardware whose primary purpose is to obtain information from or induce effects on the containing environment (i.e., as a "sensor" or "effector", respectively). An *interface device* is portage hardware whose primary purpose is to act as intermediary for coordination and sharing of relevant information among entities in its environment (i.e., among people, devices, and systems).

<———————

The entities that comprise a product may interact with external entities that operate independently or associated with other systems. The services of another system can be provided via the entities associated with a product to coordinate action and share information.

From a software perspective, a ***device*** is a logical entity, corresponding to defined services (i.e., actions and data) that it supports. A device operating in an ecosystem may be an independent entity (a product in its own right), a component of a system, or a component of a platform upon which a system operates. A logical device may be realized as a singular physical unit, as a component of a composite physical unit, or as multiple components dispersed among multiple physical units (potentially physically distributed). A "virtualized" device is software-enabled to modify or enhance the device's physical capabilities (e.g., to validate received data, maintain a log of activity, temporarily store transient data, transform data between analog and digital or alternate digital representations, monitor hardware behaviors for detecting, analyzing, and correcting device faults, or support diagnostic, prognostic, or remediation actions on device behavior).

An "emulated" device is software that approximates the observable behavior of a hardware device whereas a "simulated" device is software that approximates the internal behavior of a hardware device. Based on a specification of its expected behavior, an otherwise unavailable entity may be emulated on a computational device to serve as a surrogate. Alternatively, based on a model of its behavior, a device can be simulated for the purpose of analyzing and understanding its properties and sensitivity to external influences and to compare how variations in its modeled behavior cause these effects to differ. (An environment may similarly be simulated.)

In general, software operates on a virtualized computational device consisting in aggregate of one or more communicating computational devices, interacting with other devices (physical or emulated), all via an integrating hardware platform. A virtualized computational device includes certain software (e.g., an operating system, runtime libraries, translator, or virtual machine/container/hypervisor technology, referred to in aggregate as a "software platform") that allows other software to be built to operate without regard to the specific physical structure or properties of underlying hardware. Software-defined behavior can differ depending on the behavioral capabilities of its underlying software platform, which in turn depends upon the behavioral capabilities of its underlying computational devices and hardware platform.

Finally, a "computational platform" is an interconnected set of virtualized computational devices on which software operates. Three purposes for a computational platform can be distinguished: development, evaluation, and operations. A development platform is a medium within which to build a product. An operations platform is a medium within which a product operates in service of a customer enterprise. An evaluation platform is an instrumented (actual or facsimile) operations platform on which a product and a representation of the customer ecosystem operates for the monitoring of product operation to analyze its conformance to prescribed behavioral qualities.

The properties and accessible capabilities of a platform are an expression of basic physical mechanisms, possibly managed and augmented by software-defined mechanisms. The behavior of software is dependent on the aggregate capabilities and behavior of an associated set of devices operating on an integrating platform.

——————->

Terminology, Type 4

Products and Components

A ***product*** is a composite article and associated services that, applied to a designated system, induces or modifies the capabilities, properties, behavior, or information content of that system. Remotely or indirectly accessible capabilities of a product within a system may be construed as being a set of "*services*". A product is created with the intent of improving the degree to which an enabled system, within an associated ecosystem, supports a customer's objectives. A product may be created to effect changes in a single system, in alternate instances of a single system (whether operating in the same or different operational contexts), or in multiple distinct systems (e.g., having differing behaviors beyond what the product addresses).

The particulars of a product, as the medium for behavior corresponding to a coherent set of capabilities, can change over time. The ***lifecycle*** of a product is the series of versions in which the product is deployed into operational use as its capabilities

change. Each version of a product is distinguished by how it differs from that of other versions.

A product changes a system by adding or removing elements or modifying the behaviors of existing elements. A product views elements of a system in terms of their roles relative to the product's purpose. A product includes any additional hardware, software, practices, and supporting materials needed to operate effectively within the targeted system.

Components, being articles of hardware, software, data, or procedural guidance, are the raw material from which a product is constructed. Each component may be fabricated by the product developer or acquired from a provider of such articles. A product is an aggregated collection of completed, mutually-consistent components that are "operationally useful". A component may be replicated, with or without change, for use in multiple products as well as for multiple uses within a product. A product may itself be a component of a more complex product.

The set of all additional materials, used in or resulting from development of a product, are "developmentally useful" for the understanding and sustaining of the product as the customer's needs or circumstances change. These materials express the derivation of the product, including all relevant background and reference materials, specifications, data, development environment (including process, methods, and tools used), evaluation materials, and associated assumptions, issues, and rationale. During development, these materials are created and revised to provide current developers with a shared understanding of how the product is being developed. Upon completion, these materials provide future developers with information concerning how the product was developed, substantiation as to why the product was developed as it was, and perceived implications of any likely future changes in the product.

A Problem-Solution Space

A problem-solution space defines a set of problems, each characterized by the needs it satisfies, and a set of solutions associated with each problem. A product is abstractly the realized solution to a corresponding problem. If the problem as understood or its

preferred solution changes, a different product is indicated. This is traditionally accomplished by changing the form in which the product represented. However, imagining that every problem-solution pair has an already realized product associated with it, this can viewed as discarding the previously identified problem-solution and its associated product and identifying the reconceived problem-solution and its associated product.

Abstraction and the Concept of a Family

A *family* is a set of "related" things. The relationship of interest here is a perceived "similarity" in the behavior that an envisioned set of products or components will exhibit. Restating Dijkstra, programs are construed as [similar] and therefore members of a family if they are either alternative solutions to the same problem or similar solutions to similar problems. Generalizing from programs to products, the set of potential solutions to one or each of a set of similar problems represents an envisioned set of similar products; those products are the instances of a ***product family***[2].

A product family is more precisely specified by an associated ***abstraction***. An abstraction, as the denotation of an intensional set, specifies the criteria that every instance of the family must satisfy. This criteria is the basis by which all instances of a family are described as being "similar", while recognizing that they will differ from each other in other respects. Since every member of the set satisfies this criteria, any of them can serve equally well as an example of the abstraction.

A **subfamily** is a subset of the members of a family, consisting of those instances that are similar in satisfying more restrictive criteria than the family as a whole. Every family ultimately reduces to a set of unitary subfamilies, each corresponding to a single instance that is distinguishable by its abstraction-associated criteria from every other instance of the originating family.

This concept of a family has a consistent interpretation in other disciplines as well:

---

[2] (See Appendix A, "A Mathematical Formulation of a Product Family", for an extended exploration of this and related concepts.)

- In biology, a family corresponds to a taxonomically-specified set of organisms that are subjectively classified as similar, whose characteristics distinguish them from the members of other families;
- In linguistics, a language family is a set of languages that are all derivative of a particular origin language and being similar in particular respects as a result;
- In mathematics, a (parametric) family is a set of objects that are uniformly described by a parameterized equation or function, each instance being uniquely specified by the combination of parameter values that represent how it differs from other instances of the family (in geometry, as an example, a family defines a set of curves or surfaces that all satisfy a specified characteristic equation with each instance differing in shape according to the value of equation parameters).