3.4.3 Product Family Engineering

The product family engineering model specifies a product family in the form of a generalized product model. An adjunct to this model is a mechanism for deriving instance product models from the representation of the product family. The process engineering model provides support to product manufacturing projects for developing a product specification that guides reduction of the product family engineering model to derive a customized instance product model.

A product family is conceived based on a perception that its instances are substantively similar in keeping with the domain synopsis and commonality assumptions. Differences are a consequence of how variability assumptions formalized in the decision model are sufficient to distinguish among envisioned instance products. Any partial resolution of the decision model applied to the product family corresponds to a subfamily of products that are more similar in certain aspects. A full resolution of the decision model corresponds to a subfamily consisting of a set of products having equivalent behavior, there being no remaining unresolved decisions to distinguish among them. A set of equivalent products, all being effectively interchangeable as solutions to a given problem, can be arbitrarily reduced to a singleton set.

By adopting the product model form defined for representing a product in basic software product engineering (chapter 2), the product family engineering model would have the same seven facets—requirements, environment, design, analytics, components, verification, and delivery—and their associated elements. Depending on the domain and appropriate or preferred practices, model facets and elements might be added, modified, or removed.

The product family engineering model differs from the software product model in three respects: (1) it elevates focus from the project-customer level to the program-market level, (2) it provides an aggregate representation of a product family as a whole, along with an associated means of deriving instance product models from that representation, and (3) it encompasses not only software engineering but also systems and hardware engineering considerations as needed to realize a complete product.

The resolution of all specified deferred decisions corresponds to a distinguished product model, representing exactly one derivable product. If some decisions have alternative feasible resolutions, multiple product models can be derived to allow for a comparative evaluation of which resulting product is best. An incomplete resolution of decisions, due to uncertainty about some aspects of the problem-solution, describes a product subfamily model from which an interim product may be derived for exploratory use.

Representing a Product Family

A product family is an aggregate representation of an envisioned set of products having similar capabilities but differing in aspects of their observable behavior or formulation (e.g., of associated documentation or computational platform). The motivation for creating a concrete realization of a product family is to provide a medium from which customized instance products can be derived by resolving deferred decisions without repeated effort to construct any common portions.

An initial basis for representing a product family would be the product model for a notional or previously developed instance of the family—extended to represent the family by annotating it to show how deferred decisions determine differences among the product models for other envisioned instances. Typically, initial increments will focus on assimilating capabilities of previously developed products to the degree these align or can be modified to align with the domain definition and future market needs.

Decisions span a product family—each decision can influence the content of multiple elements within any or all facets of a product model and the content of each element of a derived product model may be influenced by multiple decisions. Differences in product models are expressed as content that differs depending on how deferred decisions can be resolved. This representation will suffice as the basis for describing additional valid instances that differ from the initial instances.

Instantiating a Product Family

An instance product model is derived by applying a product specification (i.e., a partially or fully resolved decision model) to the representation of the product family.

Any partial resolution of deferred decisions associated with a product family has the effect of reducing the candidate set of derivable products to a subfamily of the complete family. Further reducing this set requires the resolution of remaining unresolved decisions.

In developing a product family, individual elements could be expressed in a reusable, adaptable, or generative form (as described in section 4.1), all requiring definition of how the element instantiation parameters are initialized based on how relevant decisions have been resolved. In some cases, the instantiation mechanism used may be able to convert unresolved decisions of a partially resolved product model into decisions to be resolved during product installation or by end-users during operation of the product.

Incrementally Extending a Product Family

In practice, a product family will represent a set of products of which only a subset is practical to build. Domain management, based on market circumstances, will prioritize support for product capabilities that are likely to be most imminently needed, with plans to expand such capabilities as program resources permit.

Having defined a product family engineering model that is a sufficient basis for building products that will reasonably support current market needs, subsequent increments will focus on expanding buildable domain scope to encompass more products—adding omitted domain capabilities and improving existing capabilities to account for evolving market needs or discrepancies found in verifying the product family model.

Verifying a Product Family

Verifying a product family entails evaluating both consistency with the domain definition, particularly the decision model, and consistency among all elements of any derivable product model. Such verification should take into account the degree to which the domain scope has been limited by program management according to current market needs and available domain resources, and thus only partially supported by the product family as built.

Verification can occur in some combination of three approaches:

- Direct inspection – As a routine aspect of building the elements of a product family model, each element is reviewed as expressing expected content of instance product model elements, as consistent with other related elements, and as accommodating applicable decisions that express differences among the derivable instances of that element.
- Selective instantiation – Alternative decision model resolutions are conceived and applied to the product family model to derive instance product models. Each of these instances are individually evaluated both as a consistent and complete product model and as a proper realization of its associated decision model resolution. This form is analogous to the means used in evaluating a realized product (either conventionally or as part of DsE product manufacturing). A representative set of product models in aggregate exhibiting all domain capabilities can serve as a basis for regression testing within the project support element of the domain evaluation model.
- Family-level predictive analytics – The product family, as the realization of an abstraction, is evaluated as a whole using formal verification methods. One approach [discussed in section 5.2 as extending predictive product analytics to intensional sets] is to evaluate a universally-quantified predicate that characterizes a property that any instance of the abstraction is expected to satisfy. A predicate's quantifiers are defined in terms of deferred decisions that are sufficient to particularize the predicate to a given product. Instantiating the quantified predicate based on resolved decisions for a particular instance product gives a predicate for a property that the product is expected to satisfy. In simplest form, an applicable verification method can be used to evaluate whether the instantiated predicate is true for the corresponding instance product. More aspirationally, the verification method can be used to evaluate whether the generalized predicate is valid for all possible instance products that result from all valid resolutions of the relevant set of deferred decisions.