3.1 The Context for DsE

This section characterizes essential aspects that provide context for understanding the DsE methodology: the nature of a domain-specific approach, the duality of diversity and change, key elements of software product engineering context subsumed by DsE (program management, hardware engineering, systems engineering, and enhanced customer collaboration), and an economic framework for managing a DsE program.

The Nature of a Domain-specific Approach

The objective of a development program is to build products for a targeted market. A program that pursues this objective by means of a domain is "domain-specific". With a domain-specific approach, efforts are leveraged by focusing on products that satisfy similar needs. A domain defines the products that can be built, both initially and as needs and circumstances change.

For a product to be viable to build and use, its capabilities must fit within the ability of developers to conceive and understand. The complexity of a product that is viable to build has increased over time as competence in building software has grown but it is not unbounded. The question then is how to decide what the bounds are for products to be viable to build.

The motivation for building a product is to enable capabilities needed to achieve the objectives of a customer endeavor. The nature and scope of that endeavor is subjectively decided. The purpose of a domain is to characterize products that will effectively address a market in which customers' endeavors and problems are similar enough to warrant similar solutions.

The Concept of a Coherent Market

DsE is conceived as a viable approach due to targeting products to a coherent market. A market is coherent if it represents customers that have similar needs. The definition of coherence is subjective but not arbitrary. The motivation is to identify an opportunity for leverage in the realization of products that are a good fit to each customer's needs.

The premise of DsE is that similar problems are amenable to similar solutions. A single product may suffice for all (or a subset of) customers if their needs are sufficiently similar but essential differences can require different products. The goal with a domain-specific approach is to build customized products as needed without having to duplicate most of the effort that different products would conventionally incur.

Even if some customers might be viewed as constituting a simple market, having needs that are similar enough to be met by a single product, DsE allows them to be addressed independently. No customer need receive a product that is constrained to fit other customers' differing needs. Products can be customized to each customer's needs and can converge or diverge relative to other products as customers' needs change.

Any essential differences in what customers need can be identified and individually addressed with only incremental additional effort and without repeated effort on common portions. By focusing efforts on the means and materials needed to build only similar products, effort need not be expended on redoing well-understood and previously solved portions shared with previously built similar products. Efforts can focus on exploring poorly understood and unprecedented capabilities that arise with diverse or changing market and customer needs and evolving technology.

In general, any problem will have many feasible solutions, even after eliminating solutions that differ superficially but produce equivalent behavior in actual use. When solutions differ, meaningful differences are traceable to ways in which customers' needs differ. Limiting the range of problems that can be solved to only those needed by a particular market reduces the solutions to be built to ones that are both more likely to be needed and more alike.

The Concept of a Domain

A ***domain*** is the "product" of an engineering effort to facilitate building and evolving products as instances of a product family. The objective for a domain is to realize a product family whose instances provide capabilities needed by customers in its targeted market. A manufacturing effort can use a domain to build a product that is customized to be a best fit to the needs of a designated customer in the targeted coherent market. A

domain includes both a representation of a product family and an associated means for mechanically deriving instances of that family.

For a provider, a domain is a capital investment in the means of production for a family of products suited to an evolving market. A domain institutionalizes the product development competence on which that investment depends. By anticipating the future capabilities needed by customers in a coherent market, a provider can reduce the time and effort needed to deliver products responsive to those needs.

Ultimate effectiveness of a DsE program is achieved through the coevolution of a domain with its targeted market. The domain changes as the composition of the market and the needs of customers in that market change. As the domain changes, whether to meet changing needs or to provide new capabilities as technologies or provider competence changes, its capabilities influence customer and aggregate market perceptions of how their enterprises can be improved.

The Role of Developer Decisions in Product Manufacture

Development is a decision-making process by which uncertainties about the problem and its solution are resolved to determine what product to build that will best serve its intended purpose. A developer makes such decisions, some intuitively and implicitly based on past experience and others explicitly based on systematically identifying alternatives and associated tradeoffs, to arrive at a good solution.

For a product family as an aggregate view of a set of similar products, its characteristic abstraction resolves many of the essential uncertainties, eliminating any product that is not described by the abstraction. These uncertainties represent decisions that a product developer need not reconsider.

{deferred decisions account for essential differences among envisioned products, reflecting problem-solution uncertainties and market diversity.}

Remaining uncertainties correspond to behavioral differences that represent how needs differ among customers. Domain engineering identifies and expresses these uncertainties as deferred decisions that the product developer must resolve in order to build a product that fits the particular needs of a given customer. Making a decision that

resolves an uncertainty reduces the candidate set of buildable products, comprising the product family, to a single best choice. Each resolved decision reduces the candidate set of products to a subset that is further reduced by resolving other decisions.

The resolution of any deferred decision can change if the problem or best solution as understood changes. Any given resolution of a decision determines specific aspects of a product model and the behavior of the corresponding product that the decision influences. If behavior that is common to the product family or a subfamily needs to be changed, the effort to make the change in all affected products is incurred by domain engineering; changing it in the product family means that it will propagate uniformly to every product to which that decision resolution applies—each product can just be rederived with no need for changes in its product specification.

The leverage that comes with a domain-specific approach is that it provides a descriptive form for resolving decisions so that the candidate set of products that can be built is progressively narrowed to a conceptually bounded problem-solution space. Done properly, two developers expressing the same problem will produce the same description and that description will have exactly one corresponding solution. Two similar problems are likely to have similar solutions but the resulting products will differ on account of essential differences in customer needs.

In some cases, different resolutions of some uncertainties about the problem or solution seem equally good but impose different tradeoffs or no resolution is entirely satisfactory. For this, a domain can support building multiple feasible solutions to be empirically evaluated to determine a best approximation to needs. This result can be augmented by determining whether the domain should be subsequently modified to enable building a product that is a better fit.

The Duality of Diversity and Change

Three causes of product change were discussed earlier (in section 1.3). A fourth cause of product “change” (or more broadly, the rationale for the existence of multiple similar products) is diversity of needs. Product change concerns needs changing over time

whereas market diversity concerns the co-existence of similar but differing needs within a market. Product change and market diversity are both manifestations of *variability*.

Product variability represents the different realizations of a product that result from problem-solution uncertainty, deficiency, and change. Every customer is susceptible to the implications of product variability.

Market variability reflects essential differences in customers' needs, including as market composition changes over time. A simple market is one in which only product variability, not market variability, is a concern. The implication of variability in a more diverse coherent market is the need to provide multiple products to address the needs of the market as a whole.

Customers in a coherent market, by definition, will have similar needs but those needs may nevertheless differ sufficiently to preclude those customers being properly satisfied by a single product. The idea of diversity is that different products are needed either because they address different needs or because they address the same needs differently. Two products are considered to be "similar" to the degree that they exhibit similar behavior (with products that exhibit essentially identical behavior being considered "equivalent"). Such diversity may exist among customers in a market or even for a single customer facing diverse operational circumstances (e.g., differing legal or regulatory constraints, computational configurations, or ecosystem compositions).

Any given product realization may be the result of either product or market variability or both as the composition of simple markets within a coherent market changes due to the needs of different customers converging or diverging over time. (Whether a particular product realization is characterized as a "version" of a previously existing product versus as a different product is only a matter of whether it is addressing the changed needs of a single customer or the differing needs of different customers. Both cases exemplify variability.)

Resolving either product or market variability can be viewed as choosing among a set of alternative realizable products. The product that best fits a customer's current needs may differ from the product that best fits their needs at some future time. Similarly, the

product that best fits one customer's needs may differ from the product that best fits another customer's needs. Both types of variability can be expressed in the form of a product family.

{domain vs product lifecycles and evolution}

Encompassing Elements of Context as DsE Concepts

With basic software product engineering, elements of context are taken as constraints imposed on how the product is to be realized. With DsE, these elements are reconceived as integral aspects of the overall effort.

Program Management

The objective of program management under DsE is to establish an appropriate alignment between the program and its targeted coherent market, conducive to a coordinated coevolution of the two. Changing market needs influence program technical efforts which in turn result in changing technical capabilities that influence the direction of further market change. The motivation is to create a sustained ability to build products that the market will need over time.

With basic software product engineering, program management initiates and oversees one or more development projects. Projects are independently managed but may be related in some way, such as building similar products for different customers or building different products for the same customer. Program management may impose the use of particular practices, technology, or assets, and particular process and product quality criteria, but may allow variations to the degree that project objectives differ.

With DsE, program management institutes a unified business, focusing on a coherent market of customers needing similar products. Product manufacturing projects are chartered to work with a specific customer (or simple market) to build a product reasonably customized to their particular needs. In addition, program management establishes a domain engineering effort as its technical agent to streamline projects' technical activities, eliminate duplicate efforts among projects, and ensure that products differ only as needed to satisfy essential differences in customer needs.

Hardware Engineering

Software depends on hardware for interactions with the physical world. Within a DsE approach, all aspects of product behavior are first defined as software-based capabilities that can subsequently be realized or augmented in hardware form. A specification of envisioned hardware behavior is derived to guide the acquisition or development of a suitable hardware device. Access to all associated devices will be software-encapsulated to provide augmentation of and access to physical device capabilities. Some devices, specified in the product environment model as accessible but externally provided, will similarly be software encapsulated in the product for access to the services that the devices provide.

A limited degree of hardware expertise is needed in support of a program for selection of conventional, commercially available hardware components, such as in realizing a standardized product platform. When more specialized hardware elements are needed, hardware engineering is a proper aspect of software-based product development.

Specialized hardware elements, having been specified as entities in the product environment specification or as capabilities in the product requirements specification, are specified as hardware-encapsulating components in the product design specification. The expected behavior and physical realization of each hardware element is specified in the component specification in keeping with the practices in the relevant hardware engineering discipline. The component further specifies the interfaces through which the encapsulating software is able to interact with that hardware and interfaces through which other software components are able to access its services.

Systems Engineering

Systems engineering is integrated into domain engineering with a holistic software-first approach that defines the overall product organization. This includes specifying the platform and hardware components upon which the software-based product will depend. Hardware engineering efforts are chartered as a means to more efficiently or effectively realize software capabilities and quality objectives.

{Whole Product Engineering and Manufacture; process customization in support of product customization; holistic -> uncertainty resolved at customer system level}

Hardware-software codesign determines when custom devices are needed to enable or enhance software behavior. The initial premise is to define product behavior in software with general purpose hardware and then to conceive custom devices that enable or enhance more effective behavior:

- Build the product first in software, approximating and evaluating correct product behavior, using conventional or other existing hardware and emulated versions of inaccessible or specialized hardware;
- Derive hardware selection criteria and custom device specifications as results of the software-first realization (e.g., improve speed, precision, or predictability of computations; enable or expedite interactions with the physical environment);
- Select or build specified hardware as warranted for a product that will improve efficiency and effectiveness of the customer endeavor;
- Build and validate the deployable product, having modified hardware-encapsulating software to accommodate specified hardware.

Enhanced Customer Collaboration

A product is developed to provide capabilities that support a customer in performing their endeavors. The practices that an enterprise employs are influenced by the capabilities provided by such products. For best use of a product, there must be a proper fit between that product and customer practices. Provision of a new or modified product may create the need to modify customer practices, generally enabling the streamlining and reduction of effort associated with those practices. As a result, the impetus with DsE is not to view customer practices as a constraint but rather to approach the product development as an opportunity and means to improve those practices.

The Economic Basis and Metrics for Managing a DsE Program

DsE by focusing product efforts on a coherent market enables a program to standardize on the most effective solutions to a set of similar problems. This is based on an understanding of how customers' needs in that market are similar and how and why they differ. By rethinking the product development process—whether for a singular product or for each of a set of similar products, DsE enables significant improvements in productivity over that of a basic software product engineering approach.

The objective of a DsE program is to efficiently build products customized to each customer's specific needs at reduced time and cost. This objective is achieved by creating the means to mass produce customized products, following the industrial model of organizing development into customer-focused manufacturing efforts that leverage a market-focused engineering effort. Engineering subsumes routine development work into a shared infrastructure that reduces time, costs, uncertainties, and risks of manufacturing each product as it is built and subsequently evolved over its useful life.

This benefits the program in several ways:

- Problem-solution competence is realized as a capital asset in the form of a domain;
- A standardized form and terminology is the basis for a shared understanding of customer and market needs;
- Improvements in the quality of multiple products can be achieved as a result of quality improvements in the product family;
- A standardized manufacturing process results in more predictable schedule and cost estimates;
- The time and effort to build each product is reduced due to a streamlined process based on leveraging existing assets in the form of a product family.

A responsibility of management is having the means to measure and improve the (process and product) quality of its efforts. The purpose for this is to have an objective

basis for setting achievable goals and knowing the degree to which those goals are being met. Goals are quantitatively characterized by metrics that identify the measures that indicate progress in meeting those goals. The purpose of metrics-based management is to make timely reality-based decisions regarding future plans.

Domain Investment and Technical Debt

(leverage work across multiple (similar) products; reduce project efforts) (standardize on concise expression, focused on differences among similar products)

{technical debt vs investment in domain engineering: product family anticipation of potential change (effort on aspects of capability that may never be needed) versus an incomplete family (alternatives that have been envisioned as within the scope of a domain but are not yet supported or needed, giving only partial variability coverage and potential for only approximate fit to actual customer needs)}

{how domain engineering subsumes product manufacturing concern for technical debt as a development issue}

{relation of domain evolution to technical debt}

A DsE Metrics Strategy

The purpose of a metrics strategy is to guide management decision making with accurate and timely information. A metrics strategy for improving productivity builds on the three perspectives on process variation (defined in section 1.3):

- *Performance* – Monitoring work progress against a plan, to determine when and how to revise the plan (e.g., to defer less critical work or avoid future rework)
- *Maturity* – Comparing performance against potential productivity in order to characterize opportunities for improving performance (e.g., through additional training or mentoring or better definition of practices)
- *Capability* – Discovering opportunities for modifying the process to increase the potential for higher productivity or product quality (e.g., increased automation or investment in reusable assets)

Measurement Objectives

Just as the scope of management responsibility differs for the several levels of management for a DsE program, needed metrics differ accordingly. All levels are concerned with overall cost and schedule performance within their scope of responsibility and satisfaction of their customers with their results. Beyond these objectives, each level has more specific concerns:

- (enterprise) economic sustainability,
- (program) enduring domain-market alignment,
- (domain) manufacturing efficiency,
- (product) product fit for intended use over its useful life.

Evaluating the Economic Viability of a Domain

The economic viability of a domain depends initially on an extrapolation from an organization's experience in building singular products. The objective is to determine whether the cost of developing and sustaining a domain and then deriving each of a set of similar products over a prescribed timeframe is sufficiently less than the aggregate cost of developing and sustaining the same set of products individually.

{revise and combine Figure 3.1-1[a&b]; explain consistent with experience (change "application" to "product" and "application engineering" to "product manufacturing")}

N_A	number of applications expected to be delivered
N_{AV}	expected number of versions/releases to customers of each application
N_{AI}	expected number of iterations to produce an acceptable version/release
C_{DN}	cost of domain engineering effort not directly attributable to a project
C_{DI}	cost of direct domain engineering support for an application engineering iteration
C_{AI}	cost of each application engineering iteration
C_C	full life cycle cost for conventional application: $C_{CD} + C_{CM}$
C_{CD}	conventional development costs for an application
C_{CM}	conventional maintenance costs for an application
C_F	full domain life cycle cost: $C_{DN} + N_A * (C_{AI} + C_{DI}) * (N_{AV} * N_{AI})$

conjecture 1: $C_{CD} < C_{DN} + N_A * C_{DI} * (N_{AV} * N_{AI}) < 2 * C_C$

conjecture 2: $C_C / 100 < C_{AI} * (N_{AV} * N_{AI}) < C_C / 10$

implication: $C_{CD} + (N_A * C_C / 100) < C_F < (2 * C_C) + (N_A * C_C / 10)$

Figure 3.1-1a. A Domain Life Cycle Cost Model (from *Introduction to Synthesis* (p 32), 1990)

Baseline

- **C_P: current direct cost to build a single product**
- **N: projected number of future products**

Rough order-of-magnitude cost factors

- **Organization transition cost = C_P * 0.5**
- **C_{DE}: Total DE cost = C_P * 2.0 {?[1.0–>3.0]}**
- **C_{AE}: Product direct cost = C_P * 0.1 {?[0.5–>0.01]}**
- **Product adjusted cost = C_{AE} + C_{DE} / N**

Projected future cost

- **without DsE = C_P * N**
- **with DsE = C_P * (2.5 + N * 0.1)**

Figure 3.1-1b. A Domain Life Cycle Cost Model (from *PIr Tutorial* (p 38), 2002)