3.1 The Context for DsE

This section characterizes the nature of a domain-specific approach, the essential concepts and objectives of a DsE approach, and how DsE subsumes key elements of software product engineering context (program management, hardware engineering, systems engineering, and extended customer collaboration).

The Nature of a "Domain-specific" Approach

The motivation for building a product is to enable capabilities needed to achieve the objectives of some endeavor. The nature and scope of an endeavor is subjective. For a product to be viable to build and use, the targeted endeavor must fit within the ability of developers to conceive and understand. The complexity that is viable has increased as competence in building software has grown but it is not unbounded. The question then is how to decide what the bounds are for a product to be viable.

Building a product is at its core a decision-making process: what problem does the product need to solve, what capabilities should the product provide to solve that problem, how are those capabilities to be built, and how do we know whether the product as built has solved the problem correctly. Exploring how to resolve these decisions exposes a cascade of uncertainties and more and more detailed underlying decisions. What does someone need to know to correctly resolve these decisions?

(maximum leverage of work across multiple (similar) products; minimize project efforts)

Trial-and-error is an arduous and time-consuming approach to building a product (i.e., resolving problem-solution uncertainties). A better alternative is to look at an existing solution to a similar problem, carefully modifying that solution for a suitable fit to the current problem. The best alternative is to have developed competence in how to solve problems of a particular type, including how and why such problems are amenable to such solutions and how differences in these problems lead to differences in their solutions.

The essence of a domain is the recognition of a problem-solution space based on similarity: the problems must be perceived as reasonably similar with potential solutions that are correspondingly similar as well. Because the problems and their solutions are similar, many decisions can be resolved once for the whole set of problems. Decisions that cannot be resolved in this way are those that distinguish among customers' differing needs, to be resolved in collaboration with each customer to derive a customized solution.

The key challenge in building a product is describing the behavior that the product is meant to exhibit. This must first be in some form of expression that people can create and revise until agreed to be sufficiently complete and valid. This description must then be translatable into a form that computational hardware can enact. Traditionally, the description of intended behavior has been unconstrained in its form but written from the perspective that readers will share some fundamental knowledge concerning the nature of the problem the the product is addressing. Any information not held in common then has to be adequately explained for the rest of the description to be properly understood. The dilemma with this is that the description can become overly long and complex, making understanding more difficult and time-consuming. In practice, deciding completeness, consistency, and correctness of such descriptions has never worked very well, resulting in products that are not a proper realization of needed behavior.

An answer to this difficulty is specialization. Experience in building software has shown that problems having similar descriptions are likely to be susceptible to similar solutions. When a type of product has once been built, particular in more than one instance, developers gain insight into what constitutes a good solution for products targeting problems of the same type. If we are able to find a shared way to describe such problems and can describe a new problem in the same terms, we then have reason to believe that the solution to this new problem will be similar to those that have worked for similar past problems.

With specialization comes awareness of essential and incidental differences among problems and solutions. As we solve problems of the same type, we discover that

similar solutions work well but not identical solutions: there are essential differences among similar problems that require modifications in their corresponding solutions. However, if we look at independently developed solutions to even identical problems, we will find that the solutions, even if they produce identical results, will have differences. These "incidental" differences arise because there are typically many different ways to express the solution to a problem.

The leverage that comes with a domain-specific approach is that we can establish a form of expression that will enable distinguishing among each of a conceptually bounded set of similar problems. Done properly, two people expressing the same problem with produce the same description and that description will have exactly one corresponding solution. The expressions of two different problems will differ in well-defined terms. The common aspects of these problems will be described separately from the ways that these problems can differ; knowledge of these common and differing aspects are sufficient to enable a developer to produce a customized solution to any expressible problem.

(results in concise expression, focused on differences among similar products)

DsE provides the means to build a differently customized product for each customer within a coherent market. There may still be customers within that market that have similar enough needs that they can be treated for convenience as a simple market but even then nothing prevents later providing any of those customers with a differently modified product if any aspects of their needs are seen to have diverged.

As a means for resolving problem-solution uncertainty, DsE allows for an incompletely specified product to be realized in alternative versions, each being a different resolution of the uncertainties for comparative empirical evaluation.

By restricting the focus to a coherent market (i.e., a single customer whose needs change over time or multiple customers that have similar needs), DsE defines a viable approach that minimizes the effort needed to build multiple customized products by deriving each as an instance of a product family that represents the aggregate of capabilities that

customers in that market need. Even if the needs of some customers could be met by a single product, DsE allows them to be addressed independently, allowing the various products to diverge as their customer's needs change without being constrained by other customers' needs. The capabilities provided by a product family can be enhanced and revised over time as the needs of the customers in that market evolve.

Essential DsE Concepts and Objectives

The theoretical basis for the DsE methodology is the concept of a product family. This concept supports consideration of uncertainty, change, and diversity as factors in the development of a product, decisions as the means of representing these influences, and adaptability as a mechanism for accommodating these influences.

Criteria for a Product Family

A product family is conceived based on the concept of *similarity*. Similarity is a perception that certain instances of a specified population of interest are alike according to criteria that distinguish them from other instances of that population. This criteria can be expressed in the form of a predicate over instance properties that identifies a subset of the containing population. Instances of such a subset are relevant for some purpose that excluded instances of the population are not.

A *product family* is an envisioned set of similar products, similar solutions to similar problems. The ways in which products are similar is expressed by an ***abstraction,*** a concept that connotes a set of instances of a product set whose properties satisfy designating criteria, in the form of a predicate expressed in terms of instance properties. An "effective" abstraction is one that has an associated predicate which applied to any item will determine whether the item is a proper instance of the family defined by that abstraction. Every instance of the product set that satisfies its associated predicate is a proper example of that abstraction. Conversely, any product that does not satisfy the criteria is excluded.

Instances of a product family, being alike in some aspects, are further distinguished from each other by properties that may differ among them. In the same way, those properties further divide the product family into subfamilies. Any subset of instances

that share other properties of interest constitute a subfamily, corresponding to a more constraining abstraction.

Each instance of a family differs from but is equivalent to every other instance of the family relative to the abstraction that the family represents. A subfamily is a subset of the instances of a family that are similar in some way, satisfying a more constraining predicate, that other instances of the containing family are not. A ***subfamily*** is a subset of the members of a family, extending the associated abstraction with additional criteria that distinguishes these members from other members of the family.

A specifically enumerated set of products, previously developed under the auspices of a particular enterprise to solve similar problems, is referred to colloquially as a "*product line*". Instances of a product line may be subjectively viewed as generally similar, even to the point of including realized instances of a product family but may also include independently realized products that either may not properly satisfy the abstraction associated with the product family or may differ significantly in implementation or behavior from other instances.

A Market, Domain, Product Family, and Decisions

The premise of DsE is that similar problems are amenable to similar solutions. In general, however, any problem will have many feasible solutions. In the case of software, different solutions may not appear to be very similar at all and yet still exhibit similar behavior in use. When solutions differ, meaningful differences are traceable to ways in which the problems differ. For a domain, limiting the range of problems that can be solved reduces the solutions to be considered to ones that have more in common.

A ***domain*** is a product family and an associated means for deriving instances of that family. The focus of a domain is a coherent market consisting of customers having similar needs. The objective for the domain is to conceive a product family that corresponds to the needs of customers in the targeted market.

The instances of a product family are distinguished by means of an associated lattice of decisions that express the ways in which included products differ.

{talk about technical debt vs product family anticipation of potential change (i.e., effort that may never be needed) and an incomplete family (partial variability coverage and variability default choices to mitigate that)}

Objectives/Benefits of a DsE Approach

The DsE approach offers numerous benefits to a product development enterprise:

- consolidating and standardizing efforts to provide customized products responsive to the similar evolving needs of customers comprising a coherent market
  -
  -
- establishing a collaborative relationship between a product manufacturing project and a customer to realize a product customized to their specific evolving needs
  - create interim versions of the product as a medium for refining understanding of customer needs and demonstrate progress
  - create alternative versions of the product reflecting different engineering tradeoffs to enable comparative analyses and empirical evaluations as to best fit against their needs
  - provide the customer with a series of revised products over time as their needs and circumstances change
- leveraging product engineering effort across projects to reduce the product manufacturing effort of each project and share improvements over products
- achieving higher quality products while reducing product verification efforts, through greater use of previously built and verified assets
- improving predictability of time and effort required to deploy a higher-quality product that meets clearly defined customer needs

- realizing a modified product that fit changed needs more quickly with less risk of defects
- eliminating redundant work due to standardization and reuse of previously used assets when creating similar products

- A shared understanding of problems and solutions
- A framework for disciplined engineering methods

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 to foster a more effective 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 continuously improving ability to build products that the market will need in the future.

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 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 customized to their specific 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.

objective: Domain-Market Alignment/Coevolution (market-domain codependency; domain vs product lifecycle)

Market focus for Product Family is domain-specific: aggregate representation for a Set of Similar Products (overly generalized products = unwieldy complexity, internal & external) (benefit of flexibility with product variability)

A domain is a capital investment in the means of production for a family of products suited to an evolving market.

Hardware Engineering

focus hardware efforts: (1) speed and predictability of a hardware implementation for critical functionality, (2) necessary for analog-digital (physical vs virtual) information transfer, (3) ??;

sw can virtualize hdw: (1) logical device implemented by different physical device plus algorithm to produce needed data, (2) logical device implemented as hybrid translation of data from other physical devices, (3) single physical device that presents to product as multiple logical devices via software, (4) physical device gets data from other devices in order to perform data fusion, (5) device captures data over time for data filtering and smoothing (signal vs noise)

uses of sw as enhancement of physical device (sensor): (1) data filtering (noise) & smoothing (sensing error in device impl eg near limits of precision; excess sensitivity), (2)

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 achieve software capabilities and quality objectives.

hardware-software codesign determines when custom devices are needed to enable or enhance software behavior; 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 (first, approximate and evaluate correct product behavior in software) (build entire product in software, build/select hdw selectively after, for improved efficiency/effectiveness of endeavor) (specifications for building custom devices are derivative of the software-first realization) (build or target hdw devices that provide best realization of defined sw behavior)

Whole Product Engineering and Manufacture; process customization in support of product customization

Extended 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.