Feature-Oriented Domain Analysis (FODA) Feasibility Study

Kyo C. Kang, Sholom G. Cohen, James A. Hess, William E. Novak, A. Spencer Peterson

November 1990

Quick recap of DE terms

- **Application:** A system which provides a set of general services for solving some type of user problem.
- **Context:** The circumstances, situation, or environment in which a particular system exists.
- **Domain:** A set of current and future applications which share a set of common capabilities and data.

Quick recap of DE terms

- **Domain analysis:** The process of identifying, collecting, organizing, and representing the relevant information in a domain based on the study of existing systems and their development histories, knowledge captured from domain experts, underlying theory, and emerging technology within the domain .

Quick recap of DE terms

- **Domain engineering:** An encompassing process which includes domain analysis and the subsequent construction of components, methods, and tools that address the problems of system/subsystem development through the application of the domain analysis products.

Quick recap of DE terms

- **Domain model:** A definition of the functions, objects, data, and relationships in a domain.
- **Feature:** A prominent or distinctive user-visible aspect, quality, or characteristic of a software system or systems [American 85].
- **Software architecture:** The high-level packaging structure of functions and data, their interfaces and control, to support the implementation of applications in a domain

Quick recap of DE terms

- **Software reuse:** The process of implementing new software systems using existing software information.
- **Reusable component:** A software component (including requirements, designs, code, test data, etc.) designed and implemented for the specific purpose of being reused.
- **User:** Either a person or an application that operates a system in order to perform a task.

Domain Analysis Products Three Phases

- **Context analysis:** The results of this phase provide the context of the domain. This requires representing the primary inputs and outputs of software in the domain as well as identifying other software interfaces

Domain Analysis Products Three Phases

Domain modelling: The problems addressed by software in the domain

- features of software in the domain
- standard vocabulary of domain experts
- documentation of the entities embodied in software
- generic software requirements via control flow, data flow, and other specification technique

Domain Analysis Products
Three Phases

Architecture modelling: This phase establishes the structure of implementations of software in the domain. The representations generated
provide developers with a set of architectural models for constructing applications and mappings from the domain model to the architectures. These architectures can also guide the development of libraries of reusable components

The FODA method gives a means to apply these products to support software development in:

- understanding the domain
- implementing applications in the domain
- creating reusable resources (designs, components, etc.)
- supporting creation of domain analysis and other reuse tool

Overview

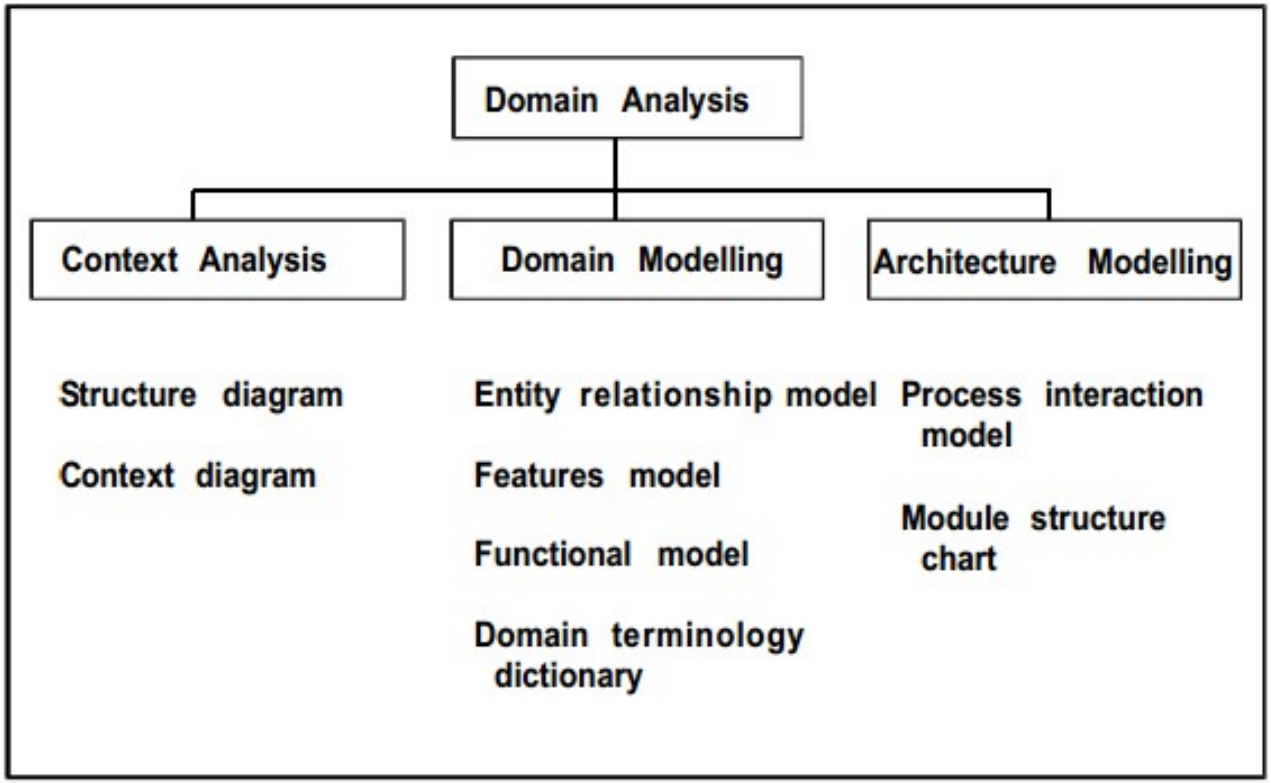


Figure 1-3: Phases and Products of Domain Analysis

Domain Analysis Methods

- **The Genesis System**
  Enormous increases in software productivity are achieved by exploiting reusable and plug-compatible modules.
  The popularized, but mythical, concept of 'software ICs' is actually a reasonably accurate description of our technology

Domain Analysis Methods

- Different methods mentioned:

- **MCC Work**
- **CTA Work**
- **SPS Work**

Overview of FODA

- Reuse at functional and architectural levels.
- Specific applications can be refined from the domain products.
- Ideal domain model and architecture is reusable from requirements to maintenance.

Overview of FODA

- Aggregation/Decomposition
- Generalization/Specialization
- Parameterization

Overview of FODA

- **Aggregation:** Abstracting a collection of units into a new unit.

- **Decomposition:** Refining an aggregation into its constituents.

- School is an aggregation of students, teachers, etc.

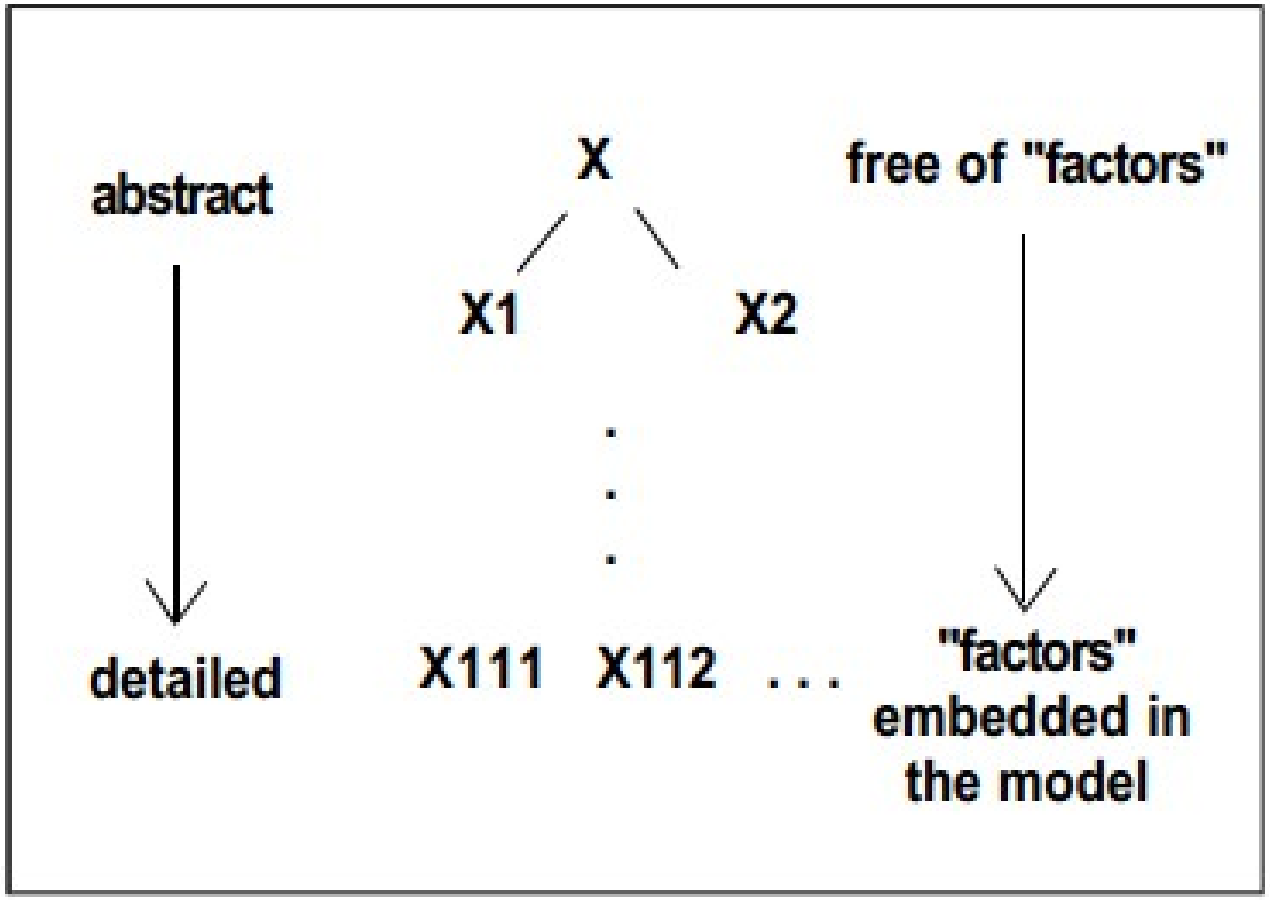
Overview of FODA

- **Generalization:** Abstracting commonalities between units, supressing detailed differences
  -> new conseptual unit
- **Specialization:** Refining a generalized unit into a unit incorporating details
- Employee is a generalization of managers, secretaries, etc.

Overview of FODA

- **Parameterization:** component development technique where components are adapted by substituting values of parameters, allowing codification of how adaptation is made within the component.

Overview of FODA



Overview of FODA

level of "factors" incorporated in a model	level of abstraction	level of reuse potential	level of productivity increase
generic (free of "factors," i.e., context free)	high	high	low
↓			
application specific ("factors" fully incorporated, i.e., context sensitive)	low	low	high

Overview of FODA

Source	Advantages	Disadvantages
Textbooks	• Good source of domain knowledge, theories, methods, techniques, models	• Reflects only specific author's views • May use idealized or biased models
Standards	• Represents standard reference model for domain	• Model may not be current with new technology
Existing Applications	• Most important source of domain knowledge • Can be directly used to determine user-visible features • Requirements documents available for domain model • Detailed design & source code show architectures	• Cost of analyzing many systems is high
Domain Experts	• Can provide contextual/rationale information unavailable elsewhere • Can be consultant during DA, validator of products afterwards	• Experts have different areas of expertise; several experts may be needed

FODA Context Analysis

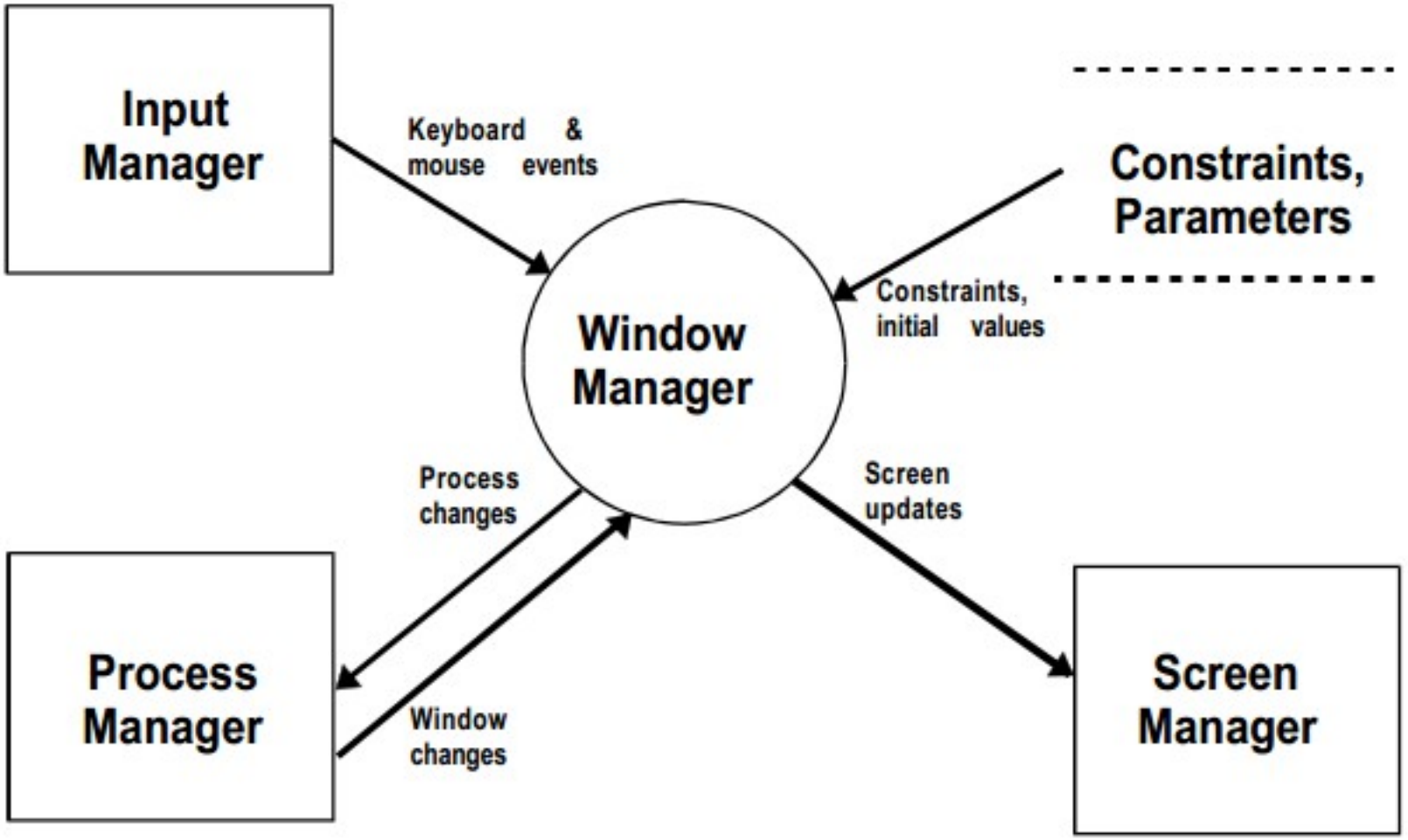
- Define scope of domain
- Results in Contex Model consisting of one or more structure and data-flow diagrams

FODA Context Analysis

Information about entities in Context Model:

- Name of the entity (an object on the diagram)
- Description of the function for a functional entity or description of the contents for a data entity
- Applicable standards and/or reusable components
- Description of variability, including the range, frequency, and binding time (i.e., compile-time, activation-time, and runtime) of the variation.
- Other items describing the attributes of the entity.
- Source for the information or for additional information

FODA Context Analysis

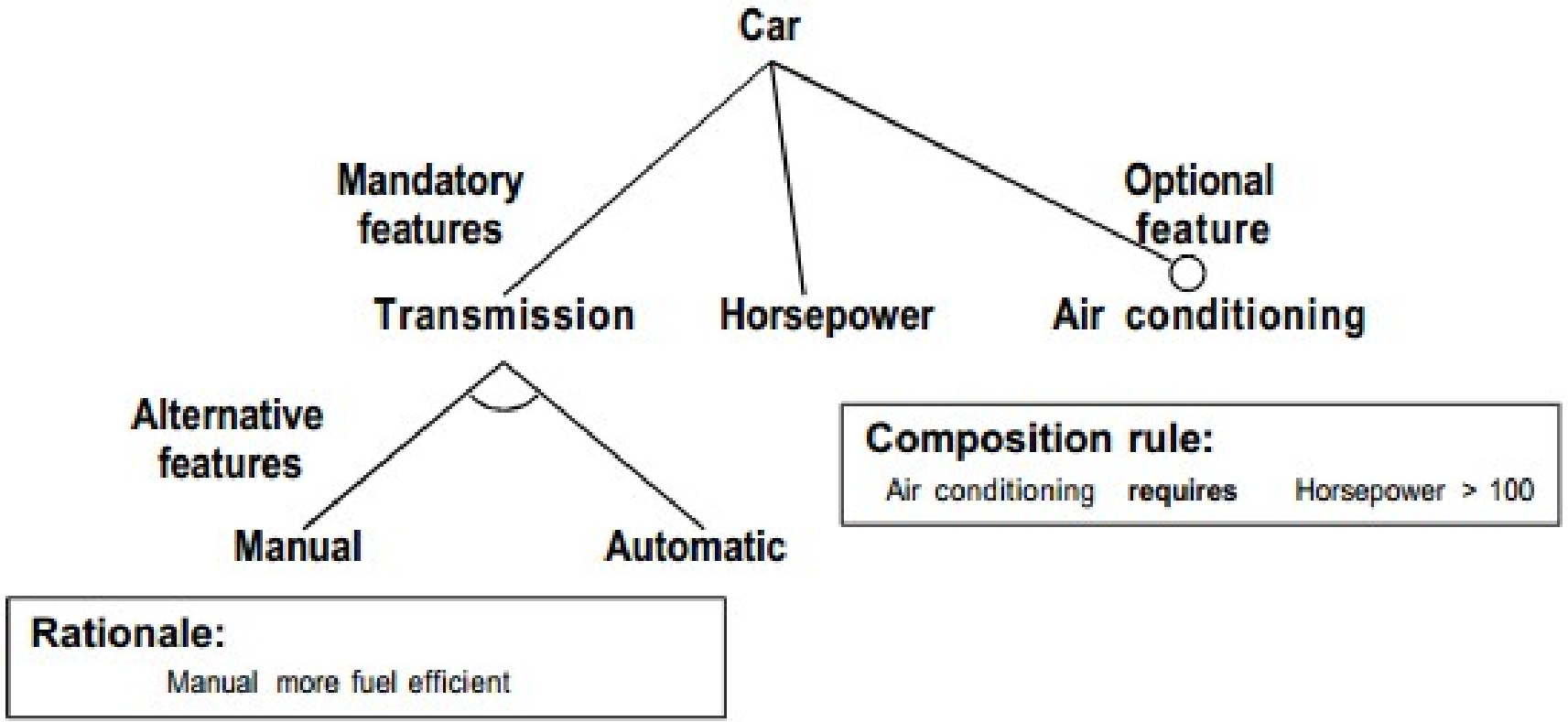


FODA Domain Modelling

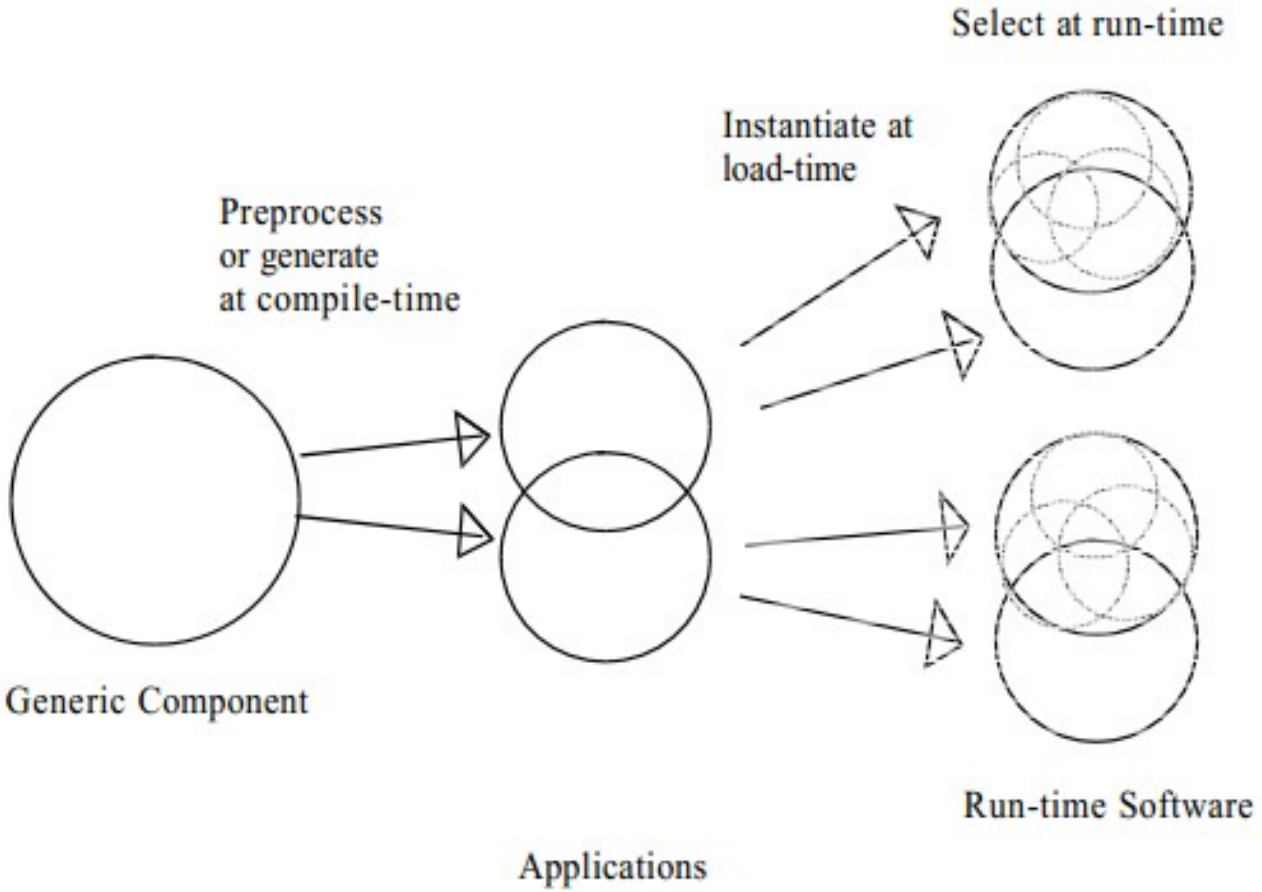
Feature Analysis:
Capture in a model the end-users understanding of the general capabilities of applications in a domain.

1. services provided by the application
2. performance of the application
3. hardware platform required by the application
4. cost
5. others

Feature Model



Processing of Features



Feature Analysis Process

1 Collecting source documents

2 Identifying features

3 Abstracting and classifying in model

4 Defining the features

5 Validating the model (by domain experts and against existing applications)

Feature Definition Form

Name: <standard feature name>

Synonyms: <name> [FROM <source name>]

One or more synonyms may be defined, and the source of each name may optionally be included.

Description:
<textual description of the feature>

Consists Of <feature names> [{ optional | alternative }]

This information shows the hierarchical structure of features, and may be represented graphically.

Source:
<information source>

This information is used to produce a feature catalog. The source of information (e.g., standards, textbooks, existing systems) from which the feature is derived is included here.

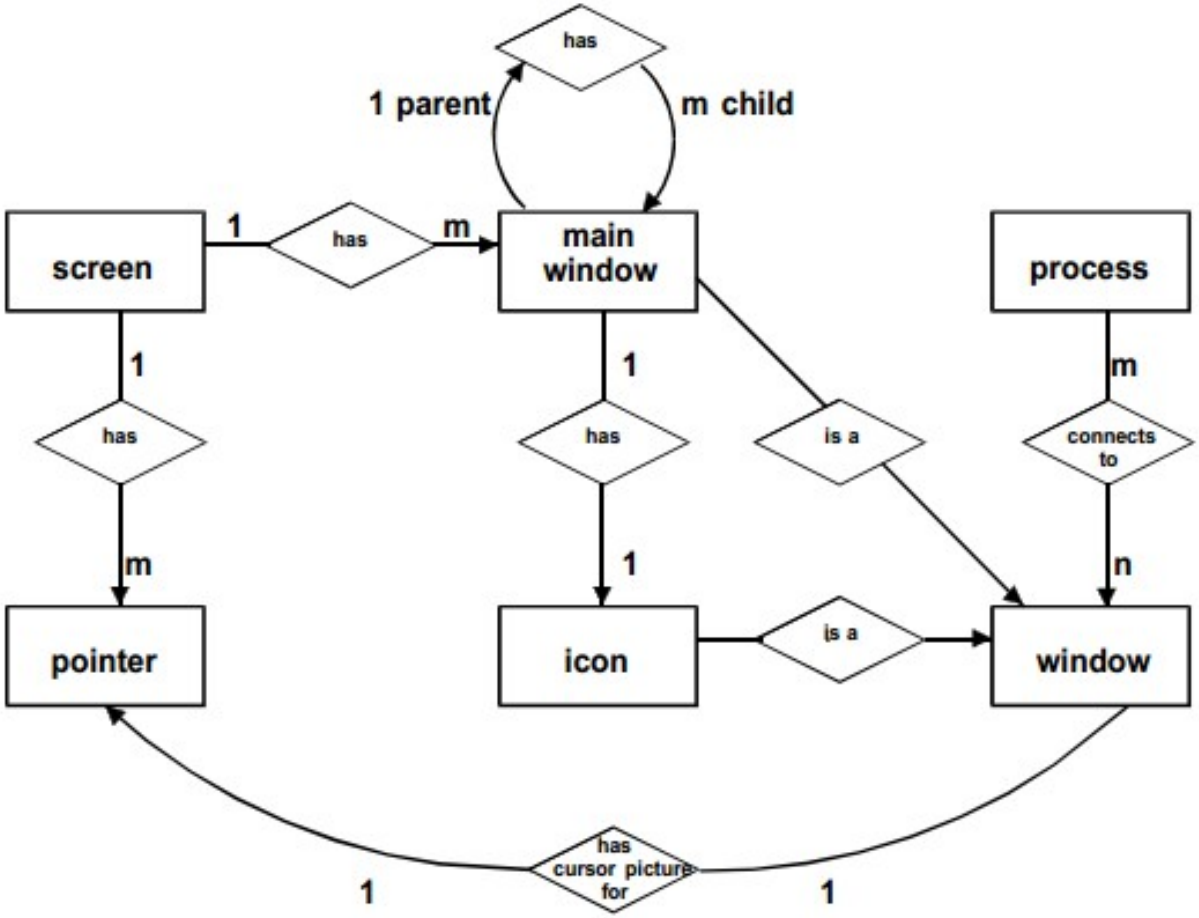
Type: { compile-time | load-time | runtime }

[Mutually Exclusive With: <feature names>]
[Mandatory With: <feature names>]

Entity-Relationship Modelling

- To represent the domain knowledge in terms of domain entities and their relationships

Entity-Relationship Modelling



Entity-Relationship Modelling

```
Entity: <entity name>

Synonyms: <synonyms>

Description:
<a textual description of the entity>

Attributes:

<attribute name>:  <value range> [<unit>]

Source:
<information source>
```

The source of information (e.g., standards, textbooks, existing systems) from which the feature is derived is included here.

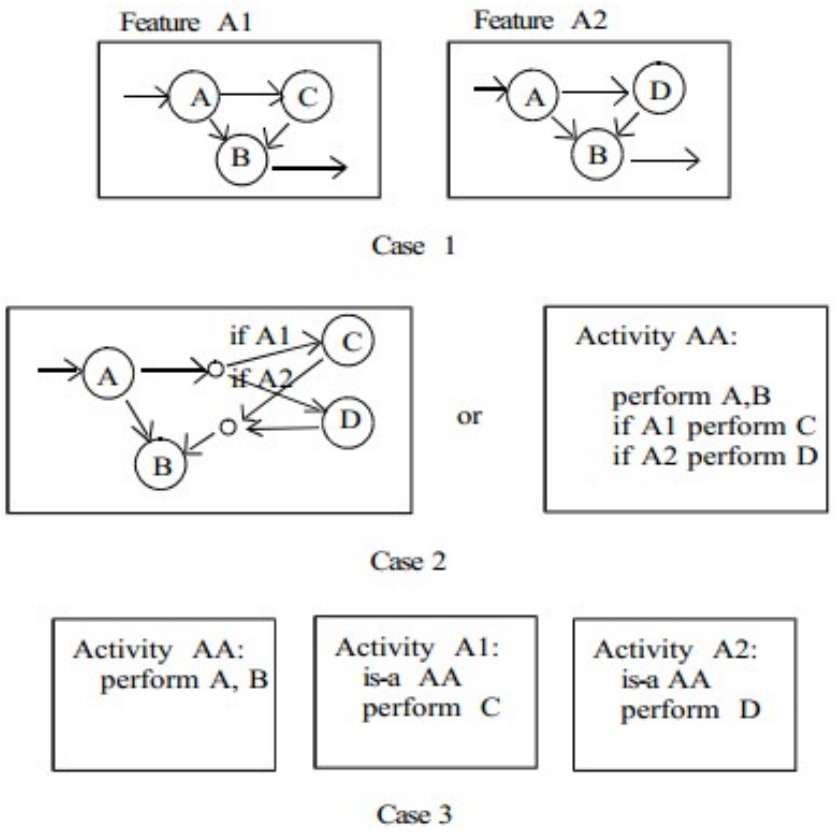
```
Relationship Type <name>

Description:

        <A textual description of the semantics of the
        relationship type. Any rules applied to the
        relationship type must also be included.>


Parts: (<role name> { <entity type name> ... |
                      <attribute name>       |
                      ANY-ENTITY}            ;) ...
```

The Parts statement defines the roles of the entities in a relationship and what types of entities can play each role. For example, the Activity hierarchy relation of Statemate can be defined as:

Functional Analysis

- Identifies functional commonalities and differences of the applications

Functional Analysis



1 Separate components for every alternative

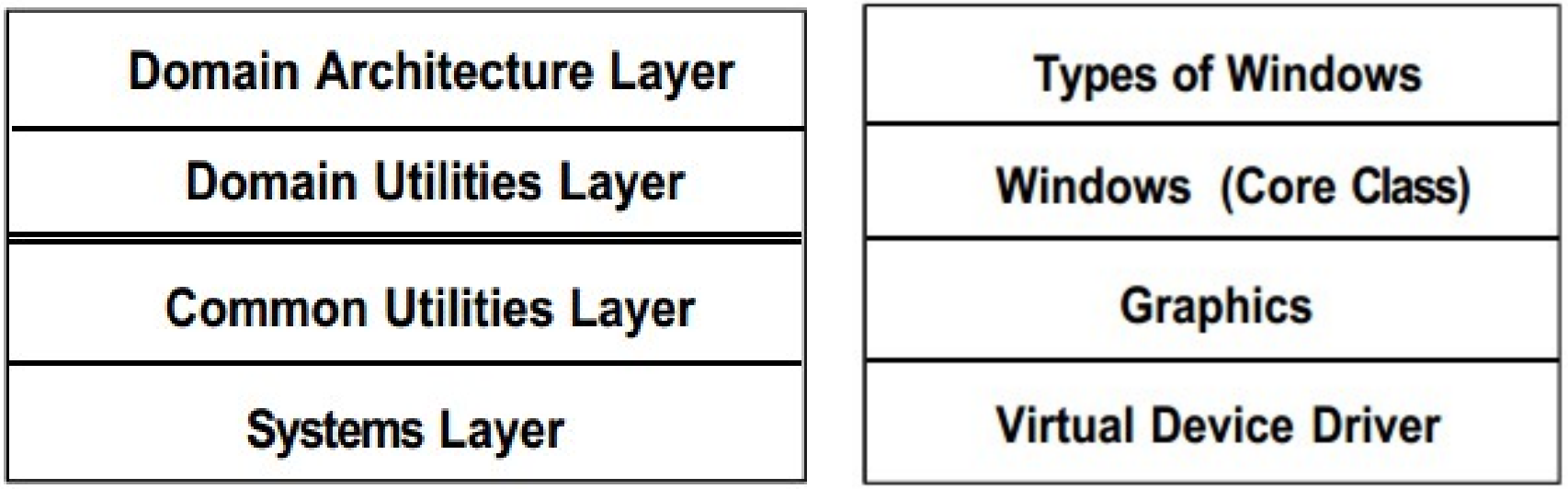
2 One component with parameterization

3 A general component, each alternative an instantiation

Architecture Modelling

- To provide a software "solution" to the problems defined in domain modelling.
- Layering to aid reuse
- Each layer is a specialization of the one below

Architecture Modelling



Discussion of the FODA Method

Main complaints:

- Lack of graphical or formalized representation (generalization/specialization in the functional level)
- Relating issues to eachother
- Could be messy if the domain is large

Conclusions

- A necessary first step.
- Provides a detailed view of the problems solved by software in a domain.
- Must take both process and products into account.
- FODA is a good basis for scoping and domain modelling.
- FODA will continue to evolve through subsequent domain analysis.

Some Examples

Issue Description Form

```
Issue: <issue-name>

Description:
<a textual description of the issue>

Raised at: <component name>
```

The "Raised at" statement indicates the component (e.g., an Activity of Statemate or a feature in the feature model) during the refinement of which the named issue was raised.

Domain Terminology Dictionary

above:	See *expose*
abstract data object:	A generic view of a widget, which allows various styles of user inputs to be transformed into simple data values
accelerator: (also called *shortcut*)	A way for an experienced user to bypass cumbersome novice commands to allow faster operation. A *keyboard equivalent* of a menu command is a type of *accelerator.*