UNIT 6

UNIT 6 DEALS ONLY WITH SALIENT FEATURES OF MICROKERNAL AND REFLECTION AS OTHER DETAILS ARE SIMILAR TO THAT EXPLAINED IN UNIT 5

ADAPTABLE PATTERNS
Design for change
Open for change
Support it’s own modification Core Adaptation to new standards, hardware platforms

The Microkernel architectural pattern applies to software systems that must be able to adapt to changing system requirements. It separates minimal core functionality and customer-specific parts. The microkernel also services and a socket for plugging in these extensions and coordinating their collaboration

CONTEXT
DEVELOPMENT OF MANY APPLICATIONS THAT USE SIMILAR INTERFACES AND BUILT ON SAME CORE FUNCTIONALITY
PROBLEM
CONTINUOUS EVOLUTION
SOLUTION
SEQUANCE DIAGRAM
MICRO KERNAL , INTERNAL SEVER , EXTERNAL SERVER, ADAPTER, CLIENT (LAYERED)

IMPLEMENTATION TABLE:

Analyse the application domain
Analyse the external servers
Categorise the servers
Partition the categories
Find a consistent and complete set of operations and abstractions
Determine strategies for request transmission and retrieval
Structure the microkernel component
Specify the programming interfaces for the microkernel
Microkernel is responsible for all the system resources

Design and implement the internal servers
Implement the external servers
Implement the adapters
Develop client applications

BENEFITS

Portability

Flexibility and Extensibility

Separation of policy and mechanism

Scalability

Reliability

Transparency

LIABILITIES

Performance

Complexity of design and implementation

ADAPTABLE SYSTEMS
1.MICROKERNAL
2.REFLECTION : PROVIDES A MECHANISM FOR DYNAMICALLY CHANGING STRUCTURE AND BEHAVIOR OF A SOFTWARE SYSTEM.
MODIFIABILITY AND EXTENSIBILITY ARE ADVANTAGES.
CONTEXT:BUILDING SYSTEMS WHICH SUPPORTS MODIFICATIONS.

PROBLEM:CHANGING SOFTWARE IS TEDIOUS DUE TO THE COMPLEX INNER STRUCTURE,
CHANGES CAN BE OF ANY SCALE,
FUNDAMENTAL ASPECTS CAN CHANGE
SOLUTION :PROVIDE META LEVEL AND BASE LEVEL OBJECTS

META LEVEL CONSISTS OF META OBJECTS METAOBJECTS ENCAPSULATE INFORMATION ,
BASE LEVEL USES META OBJECTS FOR ITS IMPLEMENTATION AND REMAINS INDEPENDENT

OF THOSE ASPECTS WHICH ARE LIKELY TO CHANGE.

AN INTERFACE IS SPECIFIED TO MANIPULATE METAOBJECTS IT IS CALLED MOP(META OBJECT PROTOCOL), PERSISTANT COMPONENTS ARE PRESENT AT THE BASE LEVEL.

DYMAMICS PROVIDE DIFFERENT SCENERIOS

IMPLEMENTATION:

1.DEFINE A MODEL OF APPLICATION(PROBLEM DOMAIN)

2.IDENTIFY VARYING BEHAVIOR

3.IDENTIFY STRUCTURAL ASPECTS OF THE DESIGN WHICH CAN BE CHANGED.

4.IDENTIFY SYSTEM SERVICES(RESOURCE ALLOCATION,GARBAGE COLLECTION,PAGE SWAPPING,OBJECT CREATION)

5.DEFINE THE META OBJECTS.

6.DEFINE THE META OBJECT PROTOCOL

7.DEFINE THE BASE LEVEL.

VARIANTS:

REFLECTION WITH SEVERAL META LEVELS.

BENEFITS:

1.NO MODIFICATION OF SOURCE CODE.

2.CHANGING SOFTWARE SYSTEM IS EASY.

3.SUPPORT FOR MANY KINDS OF CHANGE.(SUPPORTS CNAGE OF ANY KIND OR CHANGES OF ANY SCALE)

LIABILITIES:

1.MODIFICATION AT META LEVEL CAUSES DAMAGE

2.INCREASED NUMBER OF COMPONENTS.

3.SLOWER SYSTEMS.

4.NOT ALL LANGUAGES SUPPORT REFLECTION.