



MPEG Multimedia MiddleWare



MPEG-E Multimedia Middleware (M3W)

27<sup>th</sup> April 2008

## High Volume Multimedia Devices

**Application area:**  
Entertainment



### Characteristics

Proprietary

Standalone

Single Application

Single Core System

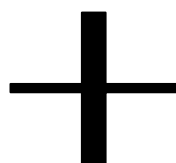
Static Configuration



## Changes in Embedded Landscape

### Application areas:

Entertainment



Home Medicare  
Automotive  
Security

### Characteristics

Proprietary

Standalone

Single Application

Single Core System

Static Configuration

Connected

Multi Application

Multi Core Systems

Dynamic Configurations

Feature-rich

**Almost 'unlimited opportunities ...  
Unfortunately also a lot of threats ...**

## M3W is split into several parts

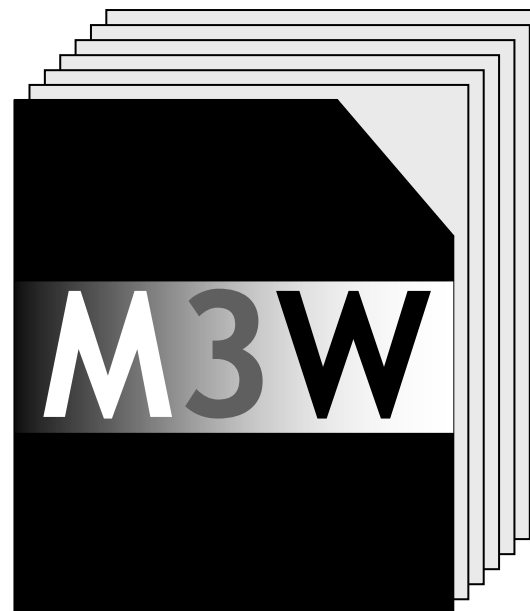
### M3W Parts:

- Architecture
- Multimedia API
- Component Model
- Resource Management
- Download / Delivery
- Fault Management
- Integrity Management

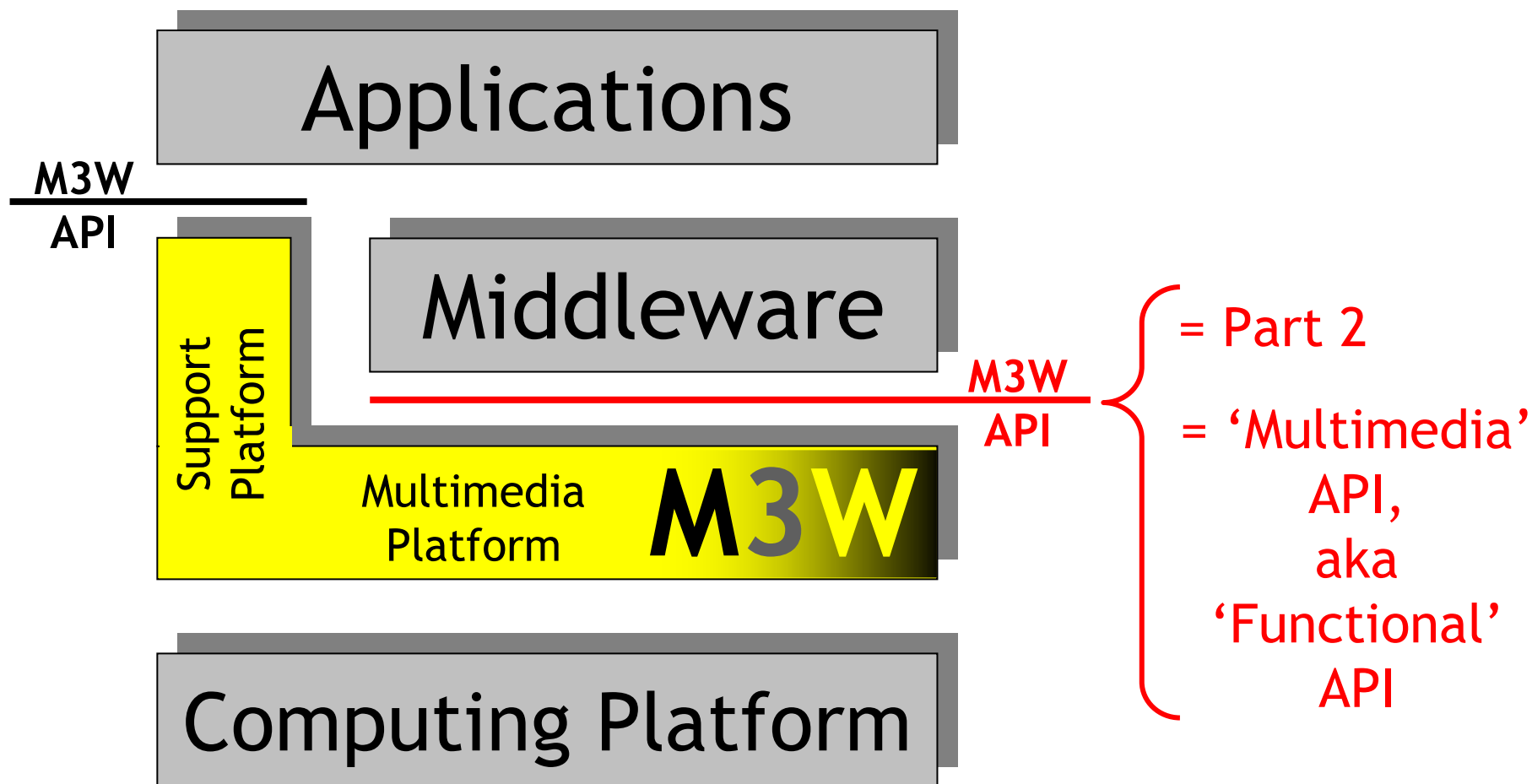


## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



## Architecture

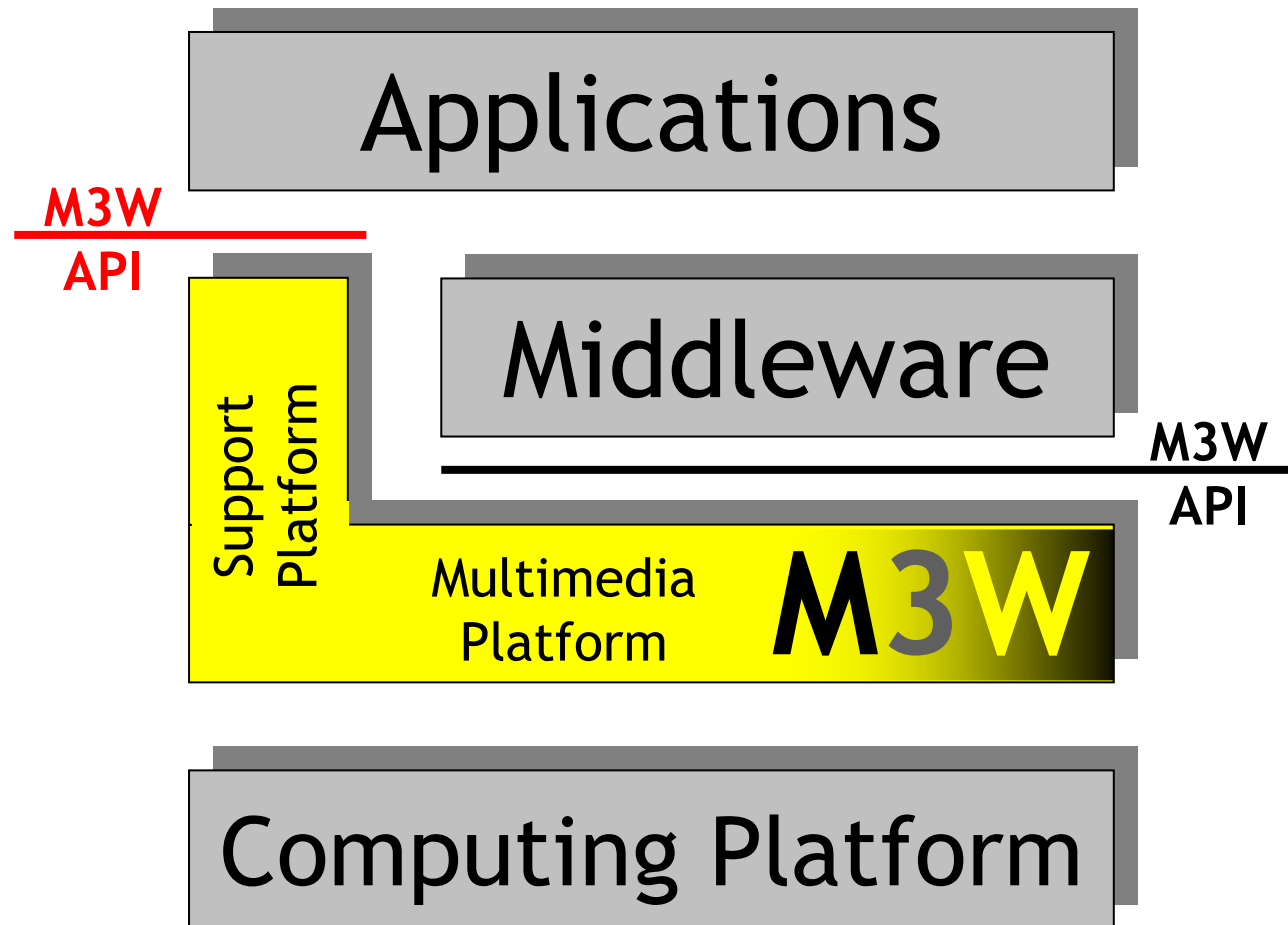


## Architecture

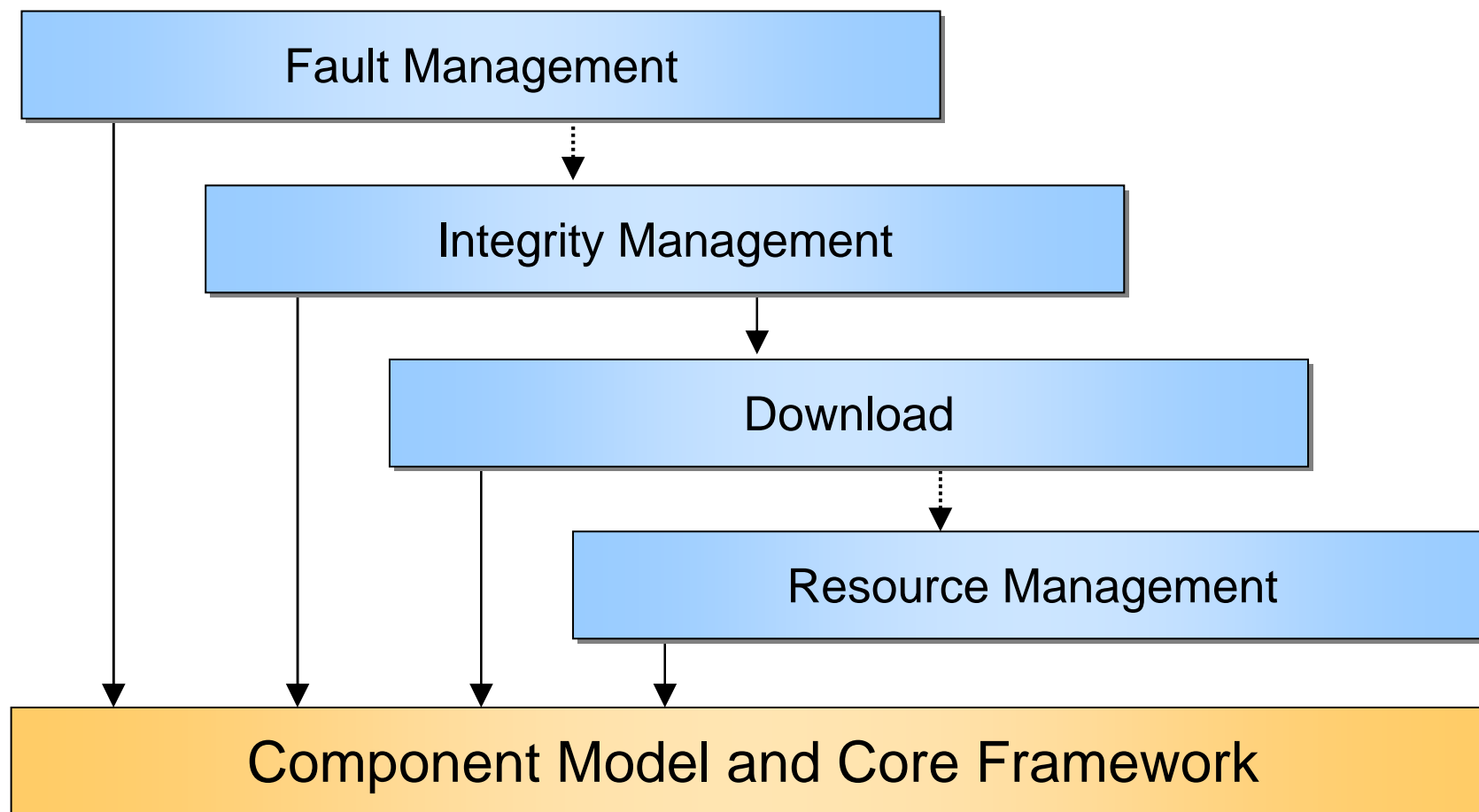
= Parts 3,4,5,6,&7

= 'Support' API,  
aka

'Non-functional'  
API



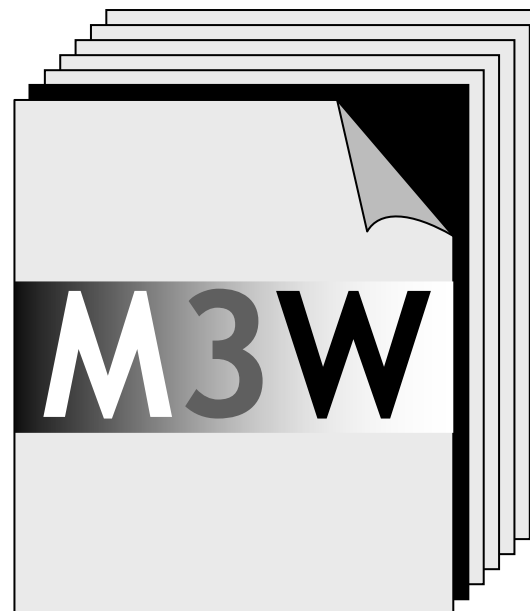
## M3W Realization Technology



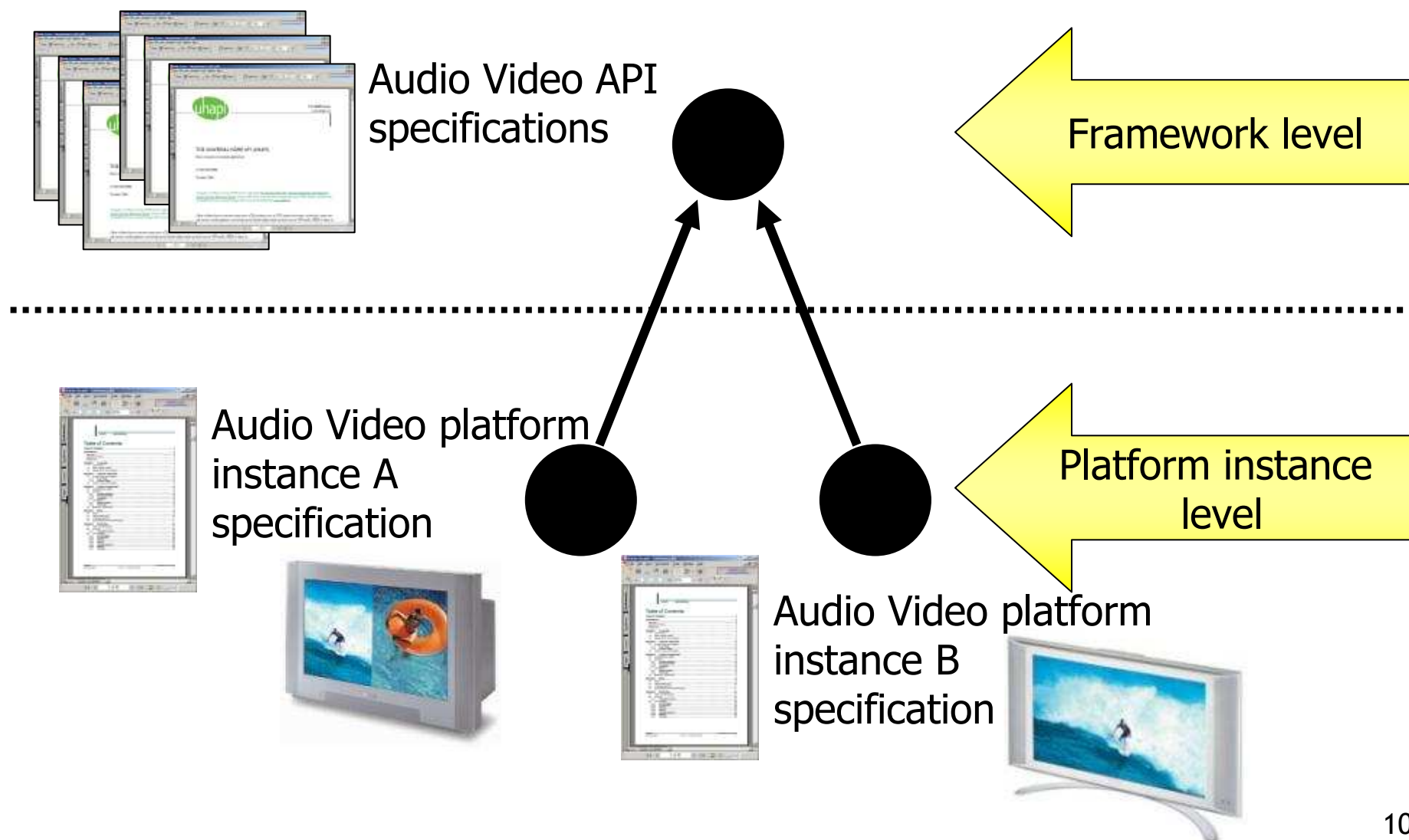


## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



## MM API does not define implementation



## **Multimedia API**

- Audio and Video API
- Governance API
- IPMP API

## Multimedia API

- Audio and Video API

- Front-end components
- Encoders / Decoders
- Video processing
- Audio processing
- Generic



Logical Components

- Governance API

- IPMP API

## Multimedia API

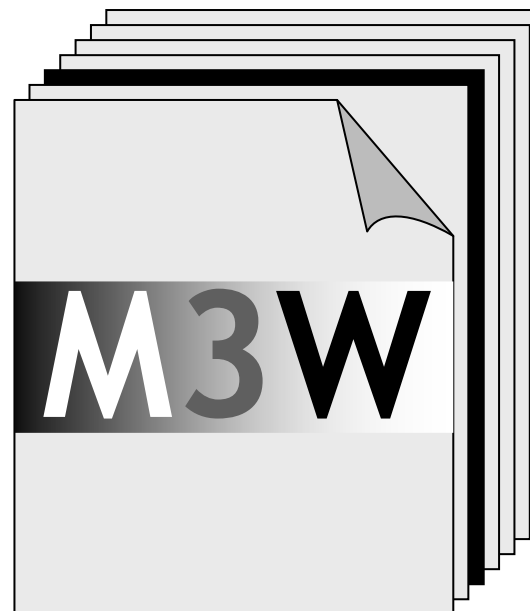
- Audio and Video API
- Governance API
  - Governance interface
- IPMP API

## Multimedia API

- Audio and Video API
- Governance API
- **IPMP API**
  - Trust Management Interfaces
    - Key Management
    - Signature Management
    - Licence Management
    - Certificate Management
  - Tool Interfaces
    - General Tool processing
    - Tool Function
    - Tool Update
    - Tool Communication

## M3W Parts:

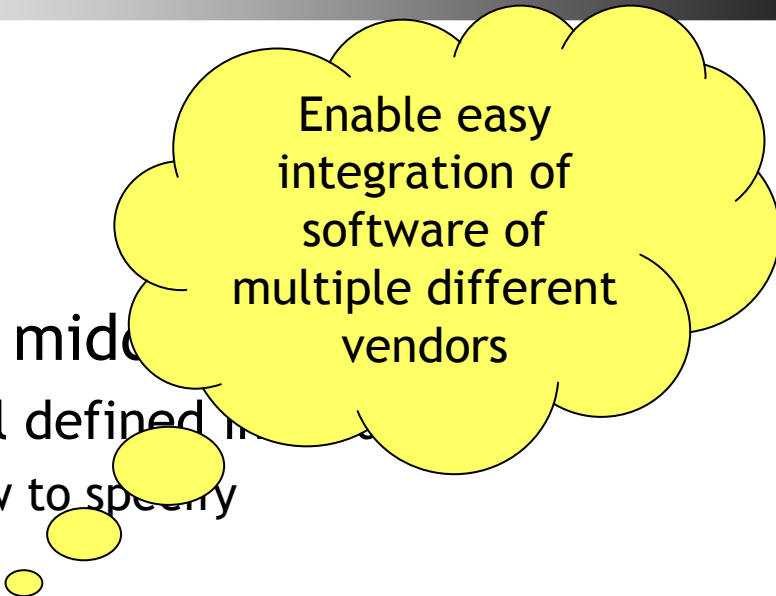
1. Architecture
2. Multimedia API
3. **Component Model**
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



## Component Model

Aim:

- Structuring the “spaghetti” middleware
  - Services interact through well defined interfaces
    - Component model states how to specify
  - Standard interfaces for
    - Instantiation of services
    - Interface navigation
    - Binding
    - Setting Attributes
    - Loading components
- Enable component trading
  - Support different views on components



Enable easy  
integration of  
software of  
multiple different  
vendors

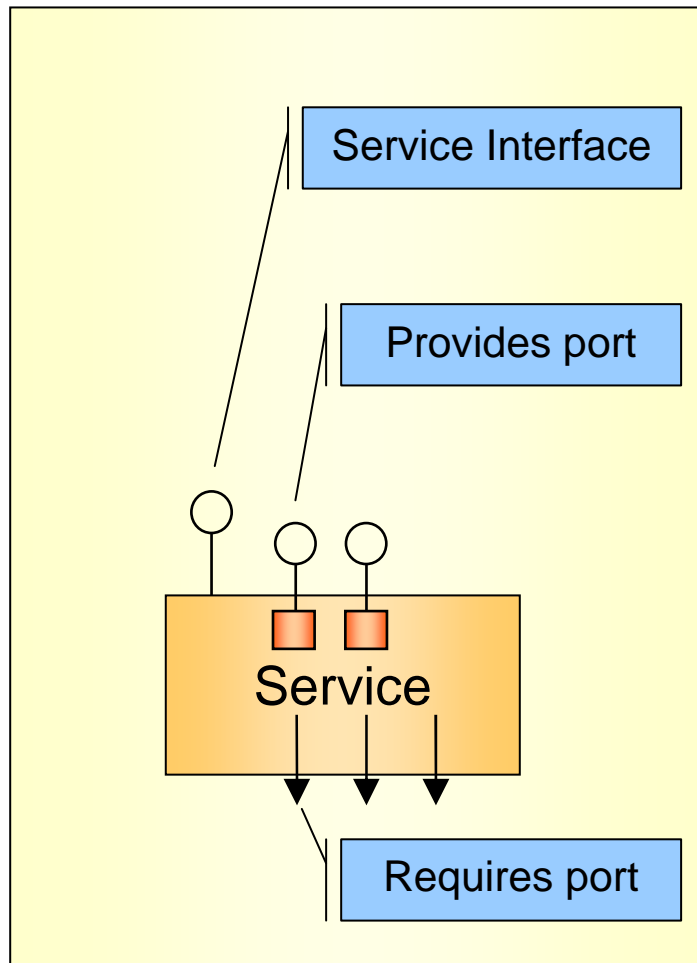


## Component Model

- Concepts
  - Service
  - Executable Component
  - Component
- Core Framework
  - Runtime Environment
  - Service Manager
  - REMI

## Intuition: Service

(Simplified ☺)

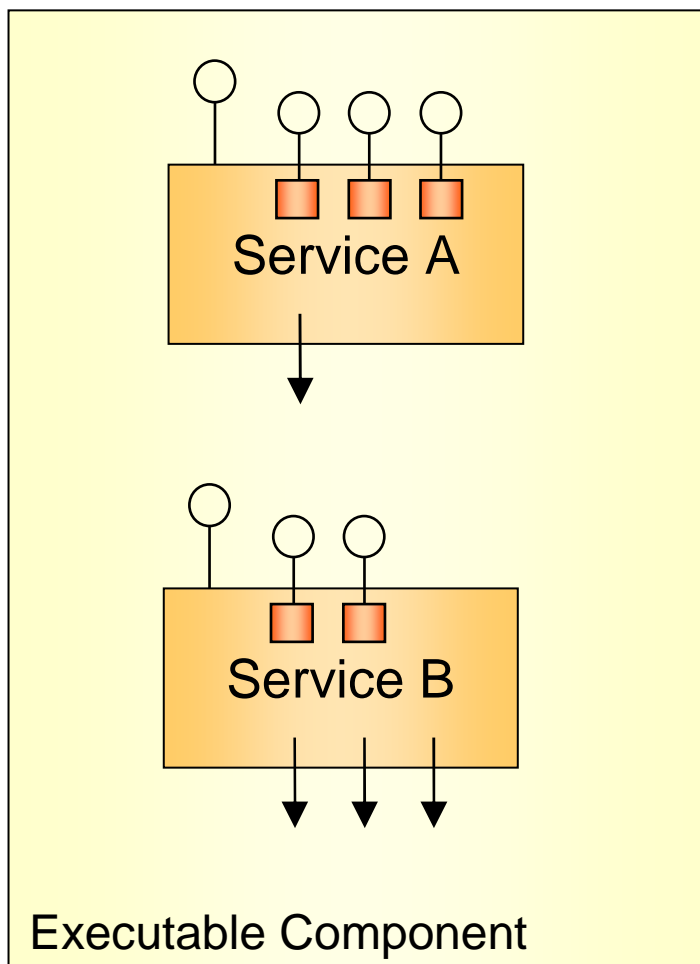


## Basic Architectural element

### Unit of Instantiation

- Service has 0 or more named Provides ports
- Service has 0 or more named Requires ports
- Ports are of an Interface type
- Interface has 0 or more Operations
- Service implements “Service” Interface
  - Obtain Interface reference to Provides ports
  - Bind Interface references to Requires ports

## Intuition: Executable Component (Simplified ☺)



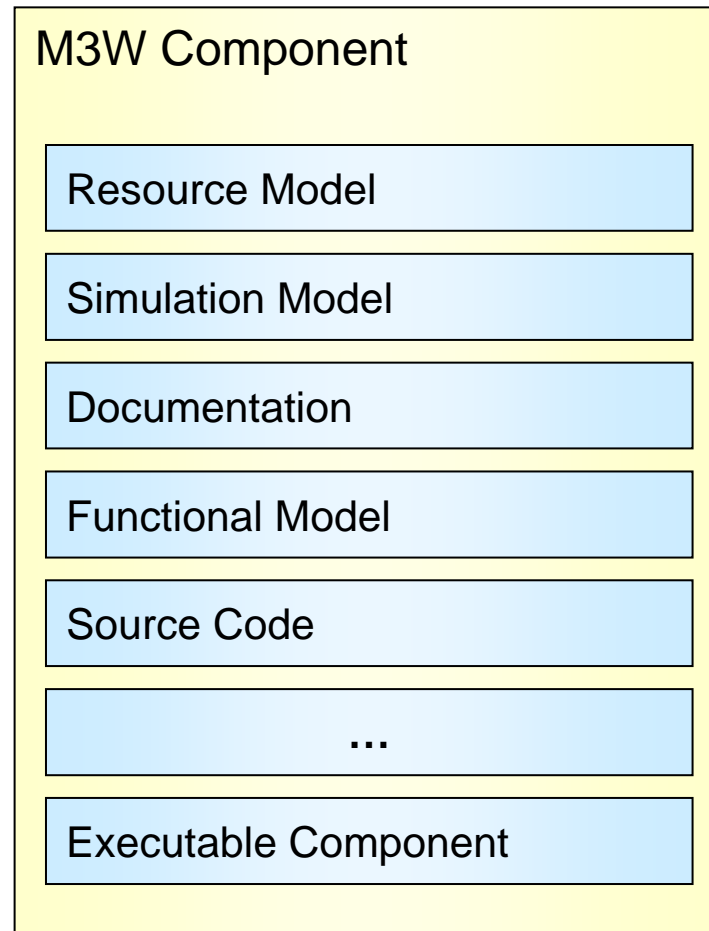
### Set of Services

### Unit of Loading

- Form depends on the OS, e.g.
  - Static in-process (LIB)
  - Dynamic in-process (DLL)
  - Dynamic out-of-process (EXE)

Also contains the “factory” logic for Services Instances, the Service Factory.

## Component



## Set of Models & Relations

### Unit of Trading

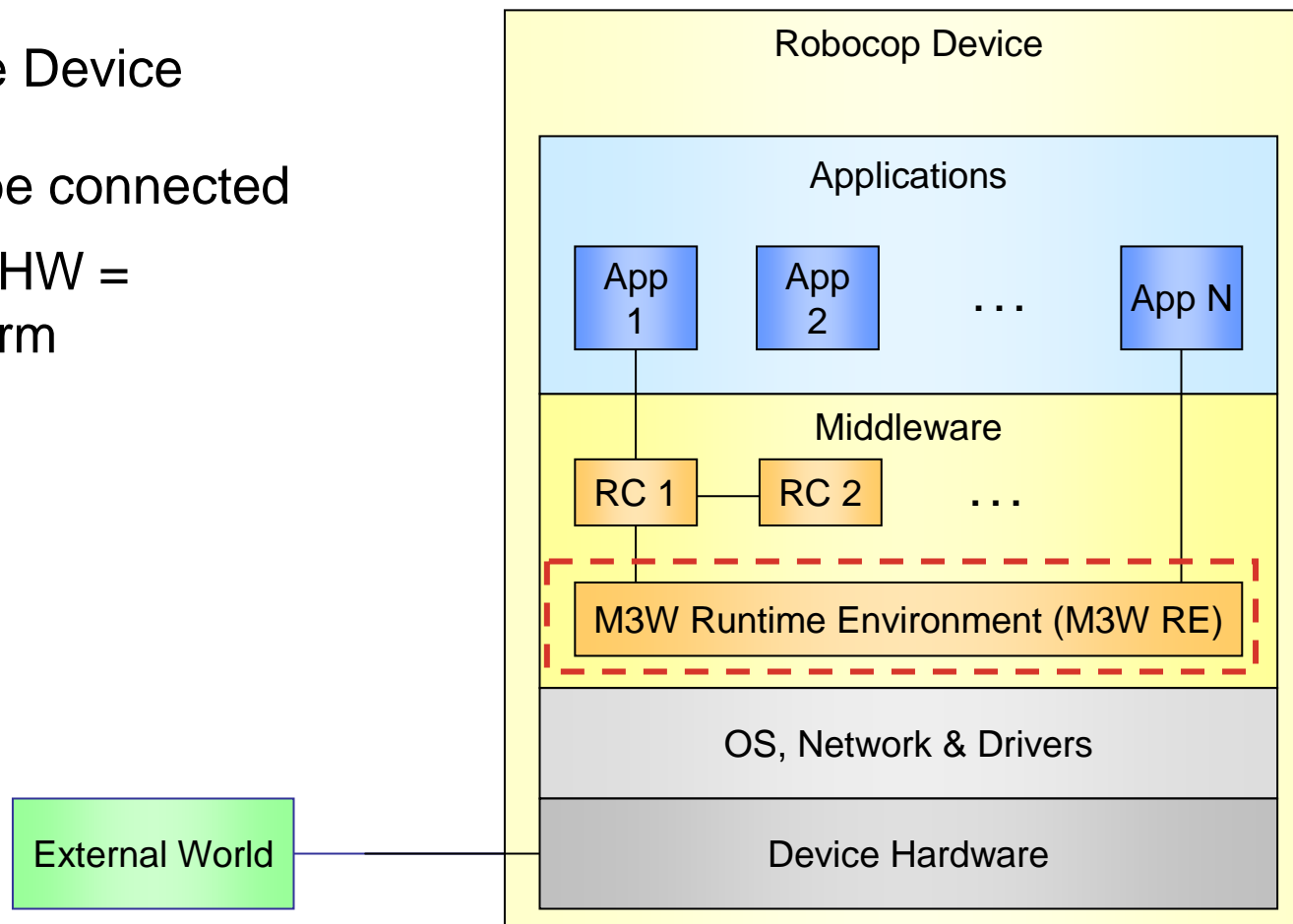
- Some model relations defined
- Typically: one (or more) of those Models is executable on a platform

## **Component Model**

- Concepts
  - Service
  - Executable Component
  - Component
- Core Framework
  - Runtime Environment
  - Service Manager
  - REMI

## Runtime Environment

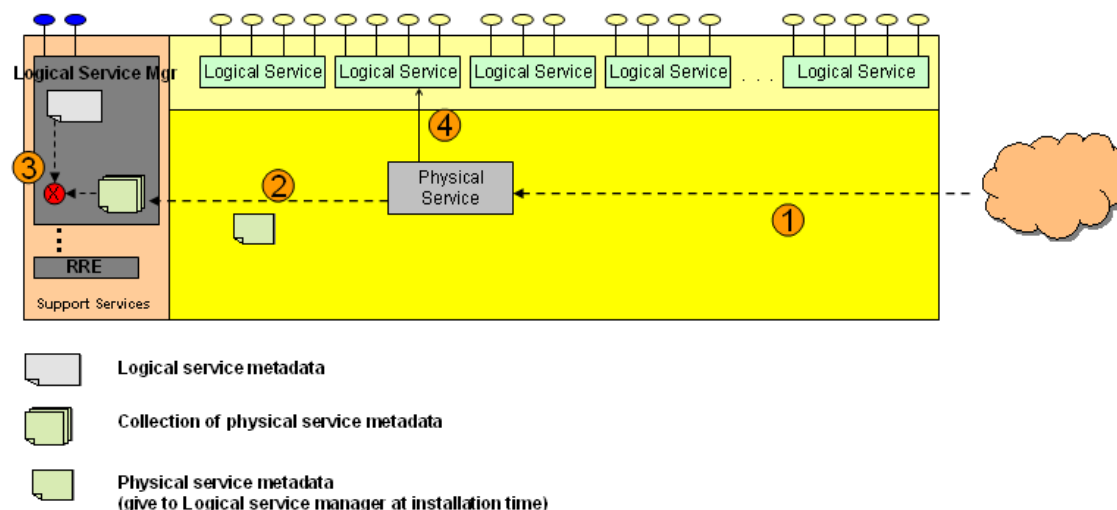
- ❖ Single Device
- ❖ May be connected
- ❖ OS + HW = Platform



## Service Manager

### Service Manager Enables:

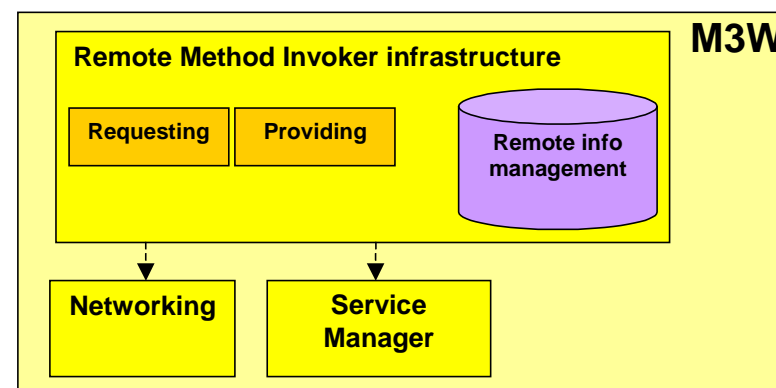
- Instantiation of realization service based on id of specification artifact
  - Based on meta data describing
    - Realization entities (services)
    - Specification artifacts (logical components)



## Remote Execution

### Remote Execution Enables:

- Dynamically discovery of devices present in the network
- Publish a subset of the registered logical services for remote execution
- Access lists of remotely available logical services on other devices.





## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



# Resource Management Framework

## Objectives:

- Robustness in the time domain
- Resource management of
  - memory, processor, bus, network, ...
- Guaranteed execution of tasks with time requirements
- Quality of Service Management

## Challenge



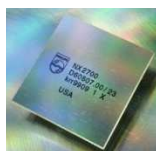
Provide robustness with respect to resource usage.

Maximize quality for the user.

Products have to be developed cost effective therefore it is not allowed to “oversize” the hardware

Systems can be upgraded and extended with new features in the field

CPU



Memory



Network



...

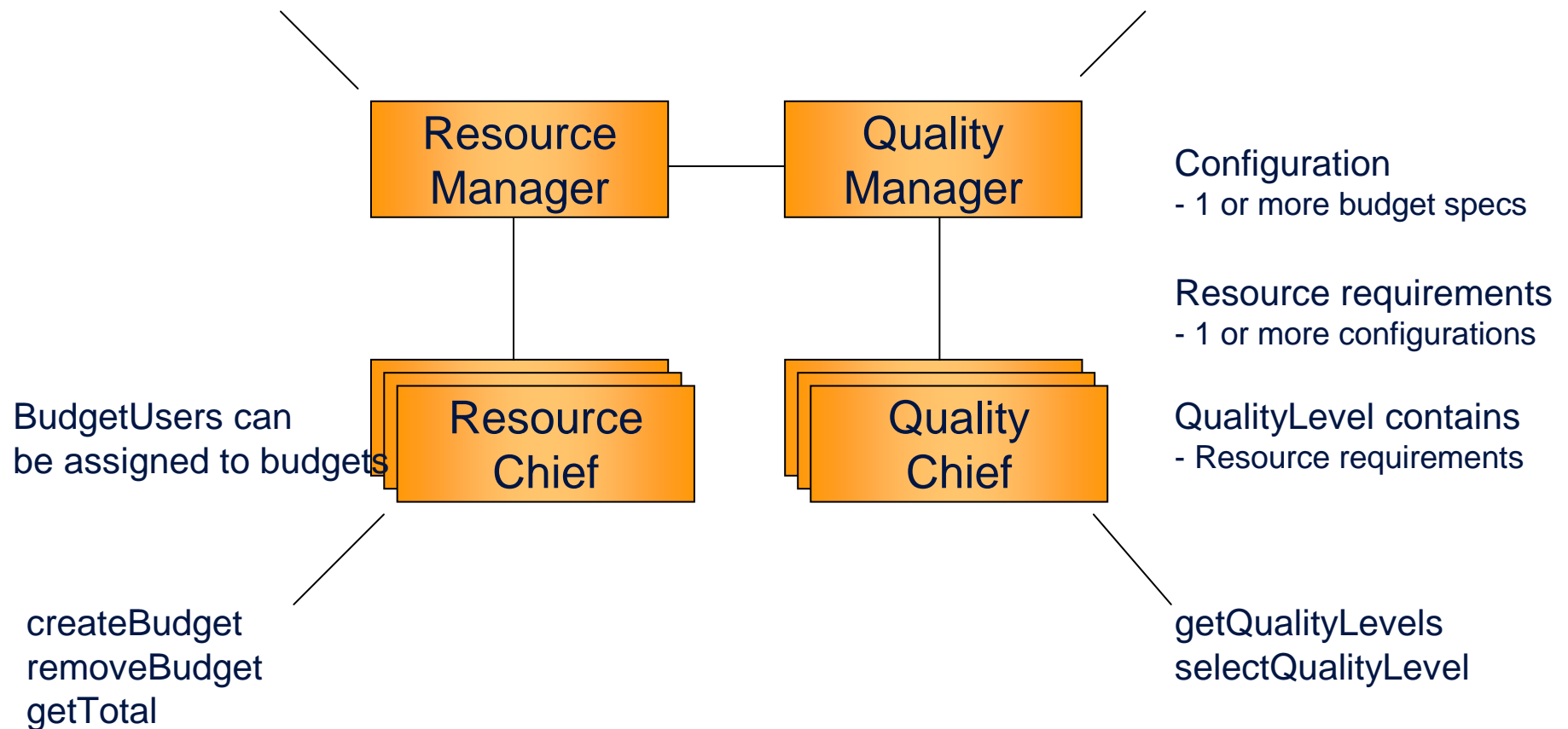
## Approach to Resource & Quality Management

- Based on a contract model
  - Resource Management Framework provides resource
  - Applications can operate at certain Quality Levels
- Negotiation based
  - Applications provide <quality level, resource needs> options
  - The RMF (selects the option and) assigns resources to Resource Aware applications.
  - *A portion of the available resources is reserved for Non-Resource Aware apps*

## Responsibilities

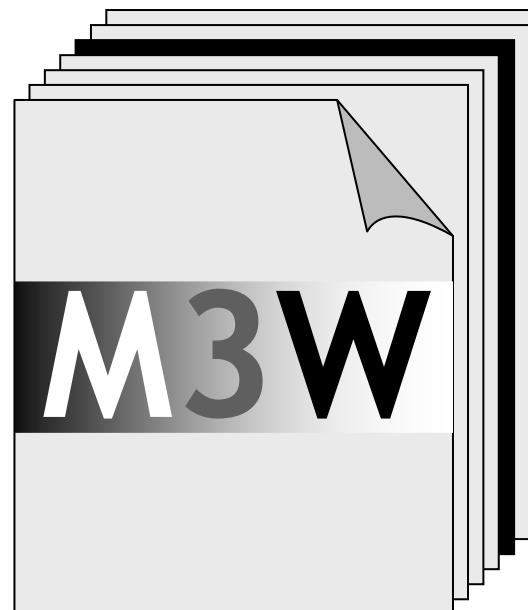
check feasibility of a configuration  
select a configuration

registration of QualityChiefs  
setting priorities of QualityChiefs  
“management”



## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



## Download Framework

- Objective: controlled download of Components
- Entities
  - **Repository**: where component to be downloaded resides
  - **Target**: device where the component will be downloaded to
- Roles
  - **Initiator**: identifies the need for a download and contacts the involved parties to initiate the process
  - **Locator**: locates Target, Repository and Decider for a download
  - **Decider**: performs the feasibility analysis for the download:  
business fit & technical fit & resource fit

## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



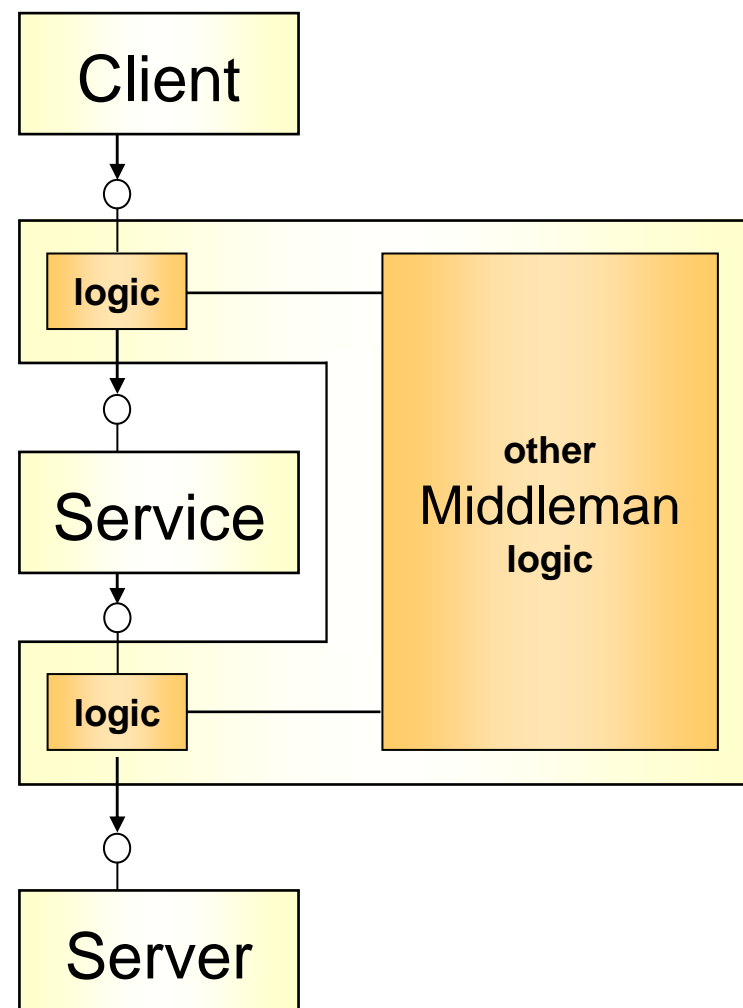


## **Fault Management Framework objectives**

- Ability to make systems composed of M3W Service Instances fault tolerant.
  - Fault-manage (instances of) “black box” Service Instances
- Be transparent to creator, server, and the Service Instance being fault-managed
- Ability to co-ordinate Fault handling between multiple Service Instances

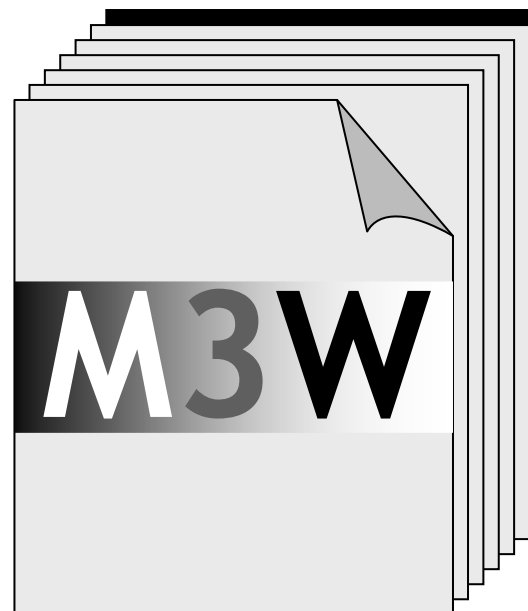
## Fault Management - Managed Situation

- Intercept creation of the Untrusted Service Instance
  - Fault Management policy
- Replace with Middleman
  - Middleman may use the Untrusted Service Instance
  - May intercept Interface Bindings
  - May contain other logic



## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management



## System Integrity Management Objectives

- Maintain a terminal in a consistent & sane state on behalf of the user and/or service provider
  - Also in the view of new information becoming available
- Manage upgrades & updates
  - Also based on context information (e.g. bus-stop)
- Provide a “reporting point” for fault management

## Maintaining Software Integrity

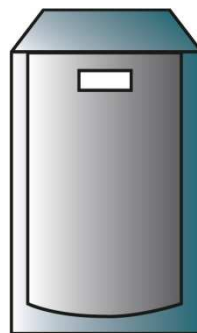
Approach: Maintaining software integrity using 3 roles !

Responsibilities: of the individual roles ...



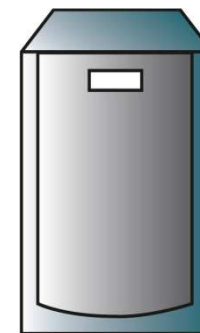
Terminal

- Externalize Model of Current Configuration
- Offer Basic Configuration Facilities



Terminal Manager

- Monitoring
- Diagnosis
- Repairing
  - Script generation
  - Script execution

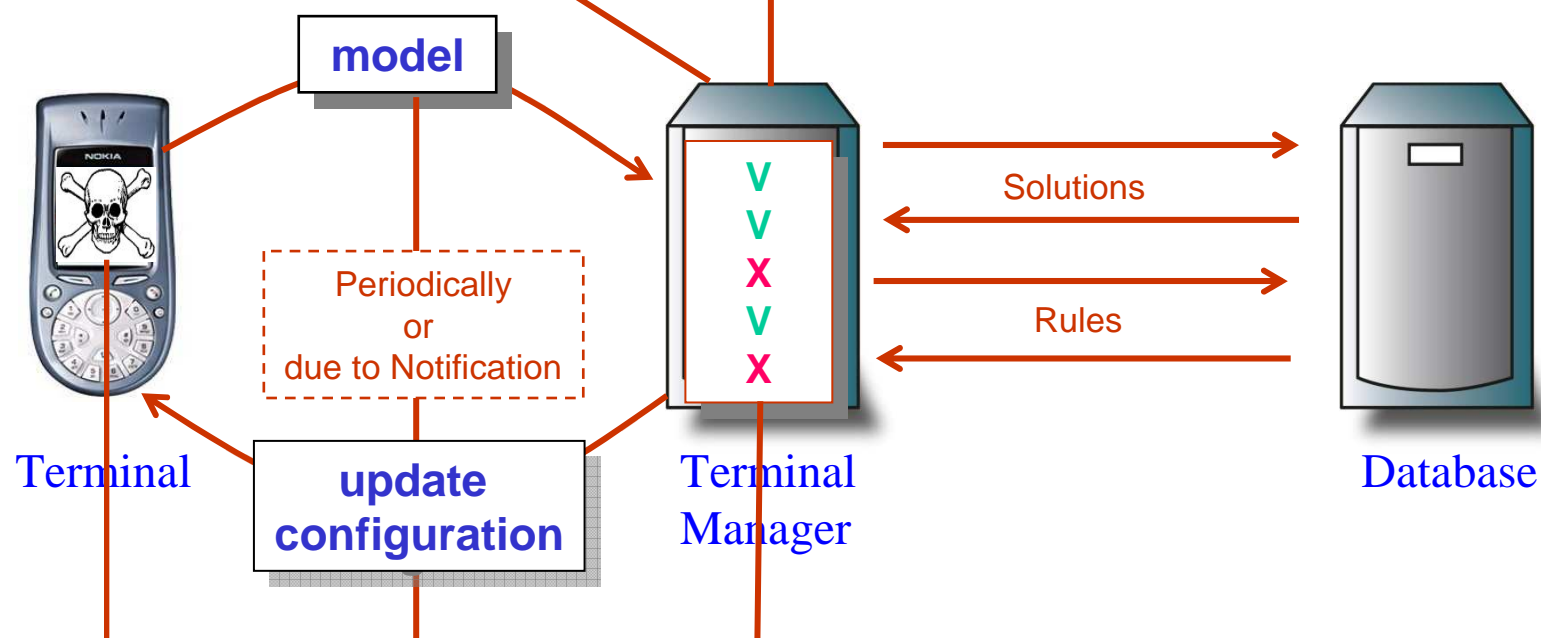


Database

- Provide rules
- Provide solutions

5. Terminal Manager will execute repair script using the basic configuration facilities offered by the terminal.

4. Terminal Manager will generate a repair script based on the outcome of the checks . This might require some knowledge from the database.



1. Terminal is not working properly

2. Model is retrieved by Terminal Manager

3. Terminal Manager will do a number of checks. These checks might require knowledge from the database

## M3W Parts:

1. Architecture
2. Multimedia API
3. Component Model
4. Resource Management
5. Download / Delivery
6. Fault Management
7. Integrity Management





MPEG Multimedia MiddleWare

