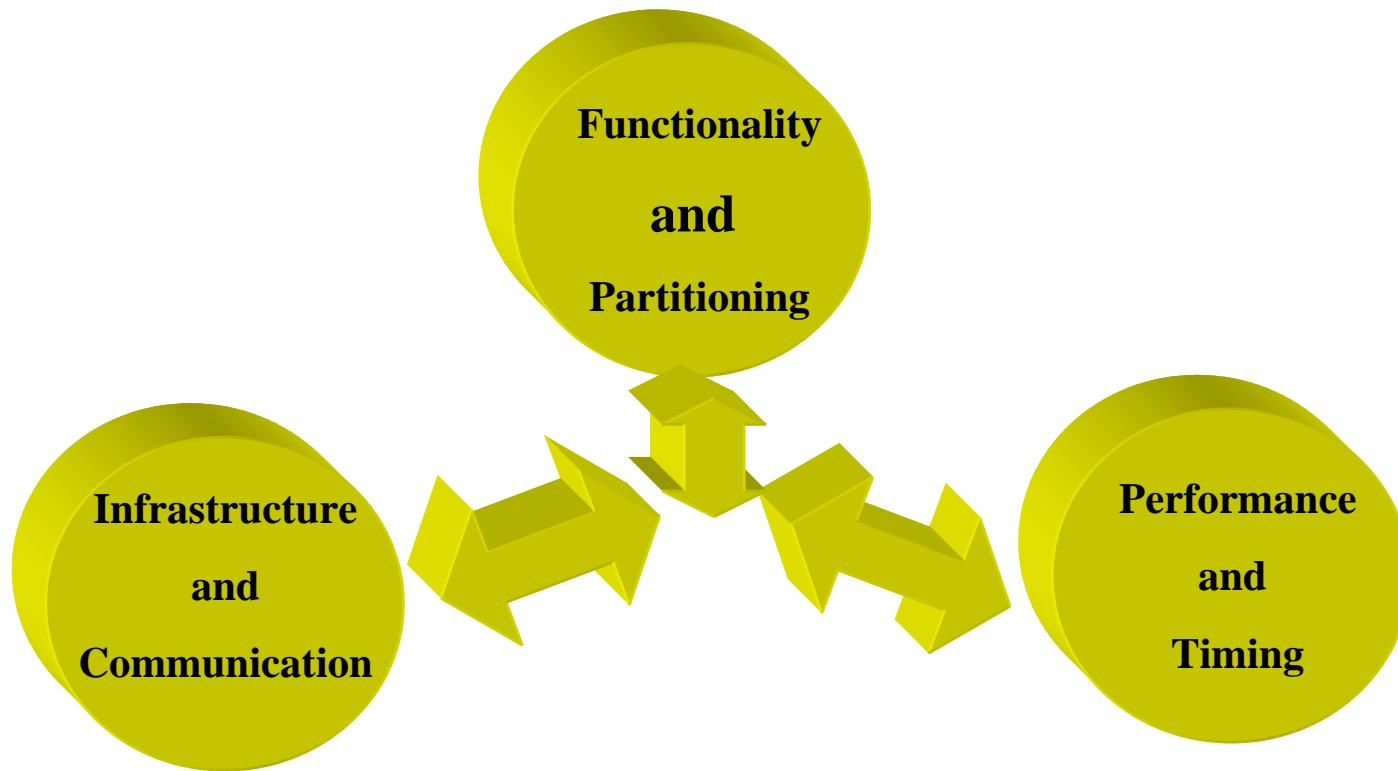


Model Driven Architecture

What Goes in the model????

**Mark Gerhardt
December 2007
mark@tripac.com**

Where are we now about architecture?



•Typical system design:

- Ideally partitions functionality
- Acquires or implements an infrastructure
- Imposes performance and timing requirements
- These are **NOT** independent choices!

Toasters

Usable Abstraction:

- Slots
- Browning Control
- Start/Stop

Side Effect Interface:

- Bendable hose with “pinlike things” on the end

Attempt at Exportable Property

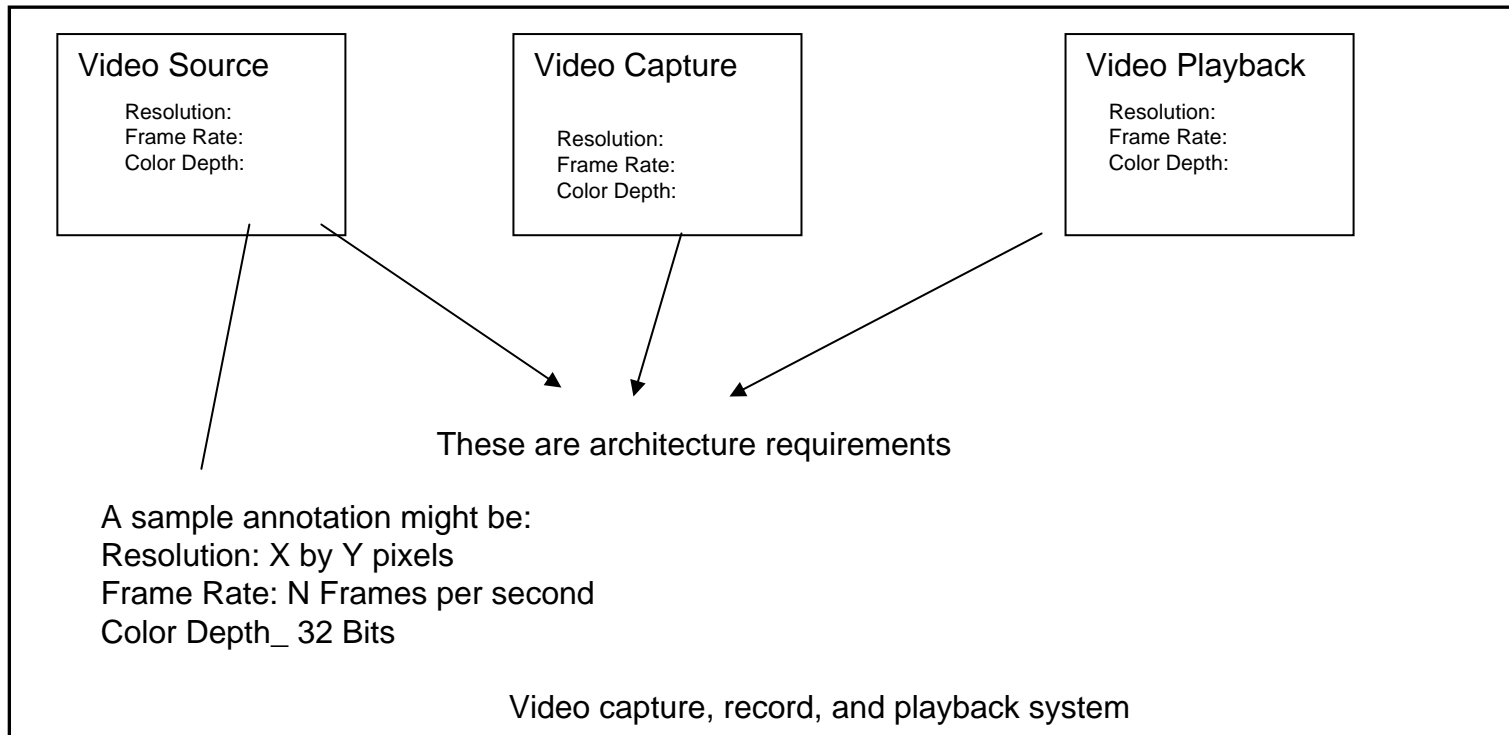
- 15A

Aggregation Rule: Left to the electrician

Video recording

Two high tech holiday presents:

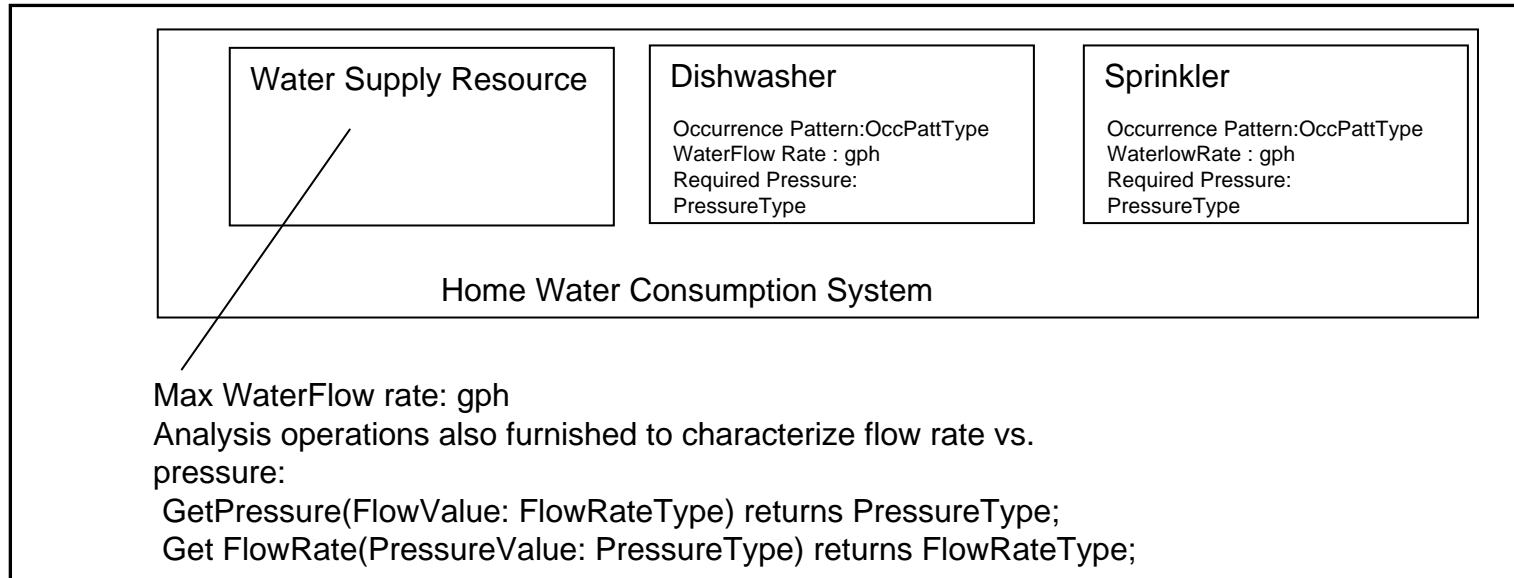
Camcorder, HDTV



Compatibility Rule: Integral multiples

Water Appliances

- Naïve consumer:
 - Sprinkler, dishwasher purchased independently



Aggregation: Dynamic determination of instantaneous flow rate checked against each invocation of any appliance. This is “pressure based” fault-tolerance in the architecture.

Family Activities

- **Functional Approach to Family Needs:**
 - **Get Groceries**
 - **Take Kids to Soccer**
 - **Buy parts and fix leaky WC**
 - **Mom also has to prepare dinner for Dad's boss by 6PM**
(Requirement: Deadline)

The Family Car - Resources

- **A Shared resource- can only be in one place at one time (apologies to Heisenberg and the rest of the quantum crowd).**
 - The result of owning one car is the sequencing of activities needing a car.
- **Resources are finite in nature and bounded in their ability to respond.**
- **Contention arises for:**
 - Allocation and Locking
 - Priority and Preemption Issues
- **Understood Examples:**
 - Safe Deposit Box Master Keys
 - A One Bathroom House with several children
 - Families with one car

Better Performance Through Concurrency

- **Shared Resource: The Family Car**
 - It can only go one place at a time
 - Actions using the car are **SEQUENCED** because they **share** the car
 - This affects performance
 - Better performance a two car family
 - Good for the economy but more difficult to schedule
- **The “Socially Correct” Implementation Binding**
 - Mom will shop for the groceries and take the kids to soccer
 - Dad will get the hardware and do the repair

The Surprise

- **The Hardware store is adjacent to the market, both 5 minutes away**
- **The soccer field is 45 minutes away through traffic!**

From a performance driven analysis, Mom should not do both the soccer task and the grocery task using the same car.

Giving Dad the job of the WC and the marketing will improve the performance of the system and allow Dad to keep his boss happy and keep his job too!

For Extra Credit.....

- **So many examples, so little time.....**

First generation CD recorders- Buffer underrun

- **How to characterize the architecture to understand the failure**
- **How to change and re-annotate the architecture to achieve current “buffer underrun protection” Why is enabling it optional??**

A More Complete Perception of Architecture

- **An architecture is a *partial implementation*.**
 - **Localized functionality to specific components and entities within the architecture structure**
 - **Has specific invocation/communication/completion protocol required by the implementation**
 - **Has important non-functional properties to add to the aggregate effect of use**
 - **Cumulative side effects- heat, capacitance, memory cache**
 - **Contention - “fighting” for data bases, devices etc.**
 - **A conforming implementation must not only provide requisite functionality, but also successfully comply with anticipated protocol and property allocations**

The Moral to the Story

- 3 **Simultaneous** Types of Choices:
Functionality, Concurrency, Synchronization
- Early system decomposition is usually done using **FUNCTIONAL COHERENCE** as the criterion for decomposition
- The domain specific requisite cooperation and collaboration between the allocated “parts” are a consequence of the partitioning.
- Resource and Communications constraints play an important role **IMMEDIATELY!**
- “**Middle Out**” development - taking temporal performance expectations into account via continued analysis - is the only practical way to assure obtaining performance relevant systems.
- “New Process”
- Layered and Deferred Implementation with aggregation rules- Motivates characterization of **offered** and **supplied** properties
- MIDDLE OUT System Decomposition – constraints, realizability, partitions, conformance all done at once.....

Your Turn

- **References.....**
 - Object Management Group UML profile for Modeling and Analysis of Real Time EMBEDDED Systems (MARTE)
 - IEEE 1471 Working Group on Architecture Capture and Representation
 - SysML Concepts
- **Sun Community Specifications”**
 - Real-Time Java
 - Mission Critical Java

- **Questions, comments, suggestions, rants.....**