

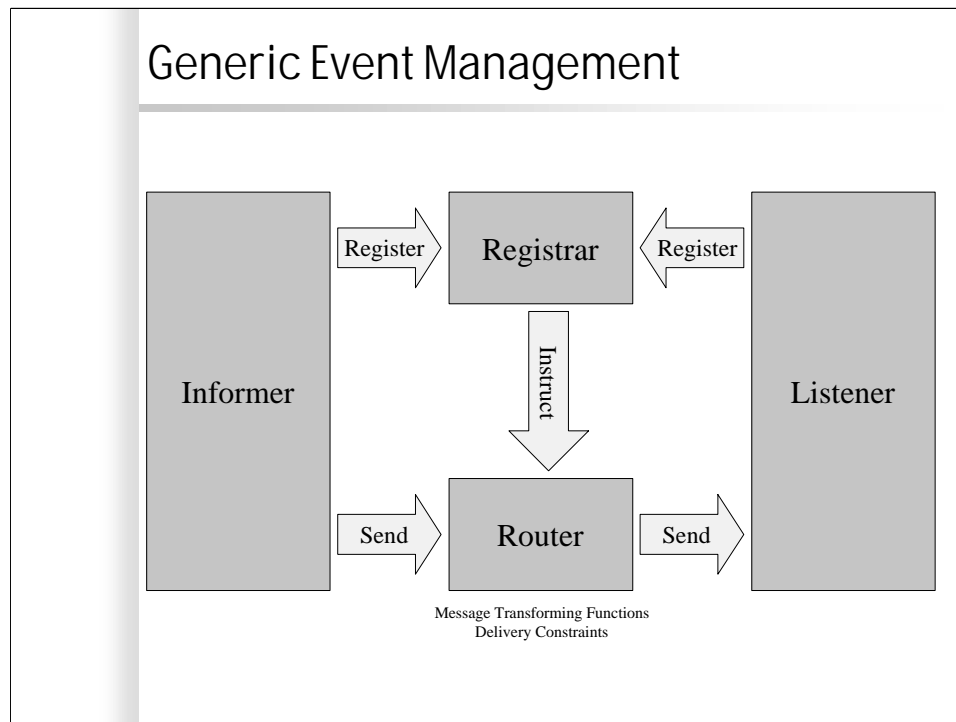
Today we're going to talk about how software is being constructed in a different way for the publishing industry. This different way is the use of component technologies and CORBA as the means of assembling these components into a complete system.

The motivation for this effort is not technology alone, it is driven by the fundamental changes in the business processes of newspapers, magazines, and other news sources.

## Why Are Events Important?

- ◆ *Distributed systems are collections of applications that collaborate to provide a capability no found in the individual components or applications.*
- ◆ *Events are used to coordinate actions between these components:*
  - Requests to perform work
  - Signals that something has gone wrong
  - Coordination mechanisms for parallel or synchronized processes

It's the value of the technology assets that drives business decisions. How can the value of these assets be increased in the presence of changing standards, increasing complexity, increasing competition, and the Internet? One answer is the better use of commercial off the shelf components and reuse of these components once they have been integrated into a system.



A standard event messaging system architecture can be seen here. This structure allows for publishers and subscribers of events to register with a clearing house to exchange location and filtering information. This is a “channel” oriented event service following the CORBA standard.

There are many options in CORBA, but we’ve used a “simple” approach where an event “object” serves as a proxy to the CORBA services to hide some of the complexity as well as some of the missing features.

## Generic Event Management

- ◆ *Informers – transmit messages*
- ◆ *Listeners – receive messages*
- ◆ *Router – allows messages to be altered and rerouted after the message has been sent*
  - Message Transforming Functions – can dynamically alter messages sent to listeners
    - Filter all messages of a specified type to a specified channel
  - Delivery Constraints – defines rules for message routing
    - All messages must arrive in order
    - A specific message must arrive in a specified time or it will be unusable
- ◆ *A common misunderstanding*
  - Events are Objects sent as message content to a channel.
  - The processing of the “object” is domain specific, therefore open ended and infinitely flexible.

The generic processes can be defined in a variety of ways, here's our definitions.

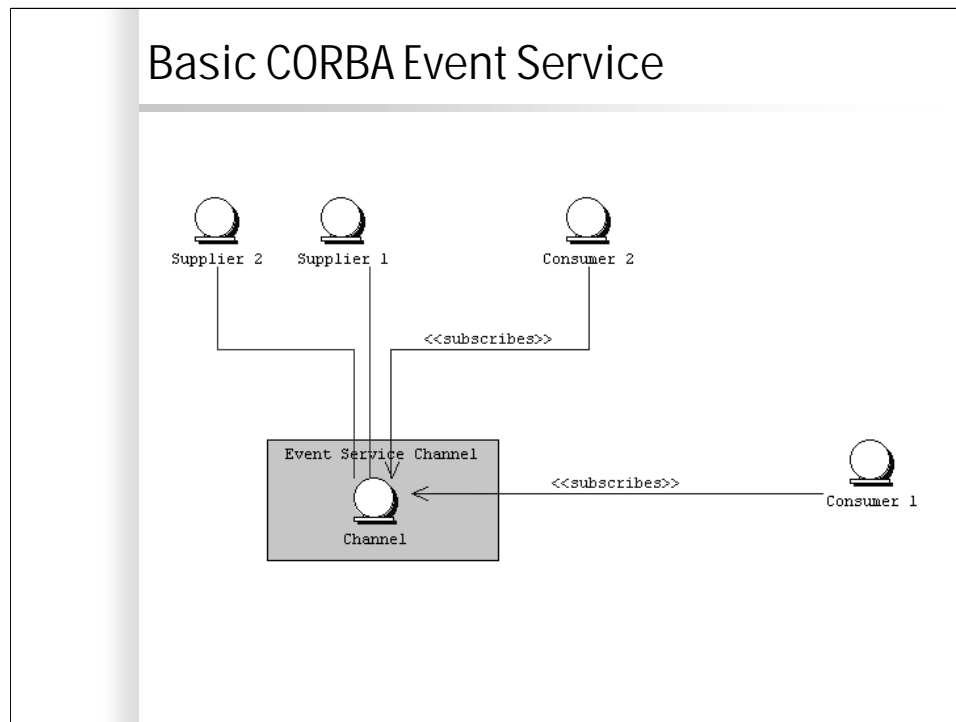
## CORBA Services

- ◆ *Traditional CORBA mechanisms are synchronous.*
  - This is actually not very useful in a distributed system.
  - In the publishing systems domain, events occur asynchronously
- ◆ *The world of newspapers is asynchronous*
  - Multiple processes collaborating in different configurations throughout the day.
  - “Page Makeup” systems have multiple suppliers and consumers as well as multiple states and status’ depending on the Point of View
    - Editor
    - Paginator
    - Production Manager

In information systems, business components are like plumbing, they connect all the important pieces of the system, provide the flow of important resources, and keeps everything going in the right direction at the right time, for the right reason.

One plumbing model is the interchangeability of parts in the North American Standard plumbing fixtures. Parts and pieces can be assembled into a final system by simply connecting each component in the proper manner to form the completed system.

This may not be the experience of the home repair person, after the 3<sup>rd</sup> trip to the hardware store for that “special” part, but it is the norm in the commercial plumbing world.



CORBA is an enabler of business components. It is not the only way, but it is one that has several advantages, and some disadvantages, over the alternatives. It is platform independent, it provides the services for:

- fault tolerance
- load balancing
- Scaling
- quality of service
- replication

These features come with a price, like anything worthwhile, but they are provided in a standard package. Like any modern system, the price is the technical competence and skills needed to assemble the system into a product. In the current business environment these skills are in short supply.

## Basic CORBA Event Service

- ◆ *Consumers and producers are completely separated from each other.*
  - Proxy objects are used for this purpose
  - The Proxy Object is obtained from the `EventChannel`
- ◆ *The `EventChannel` facilitates the data transfer between consumer and producer*
- ◆ *The Inprise VisiBroker ORB provides such a CORBA compliant Event Service.*
- ◆ *However, VisiBroker does not provide “Typed” events.*

Since the system is an open EAI based environment, tying events to specific actions and users would not be the best solution in the end.

Proxy objects are used to isolate event producers and consumers from each other.

Event channels provide the way to further partition the events.

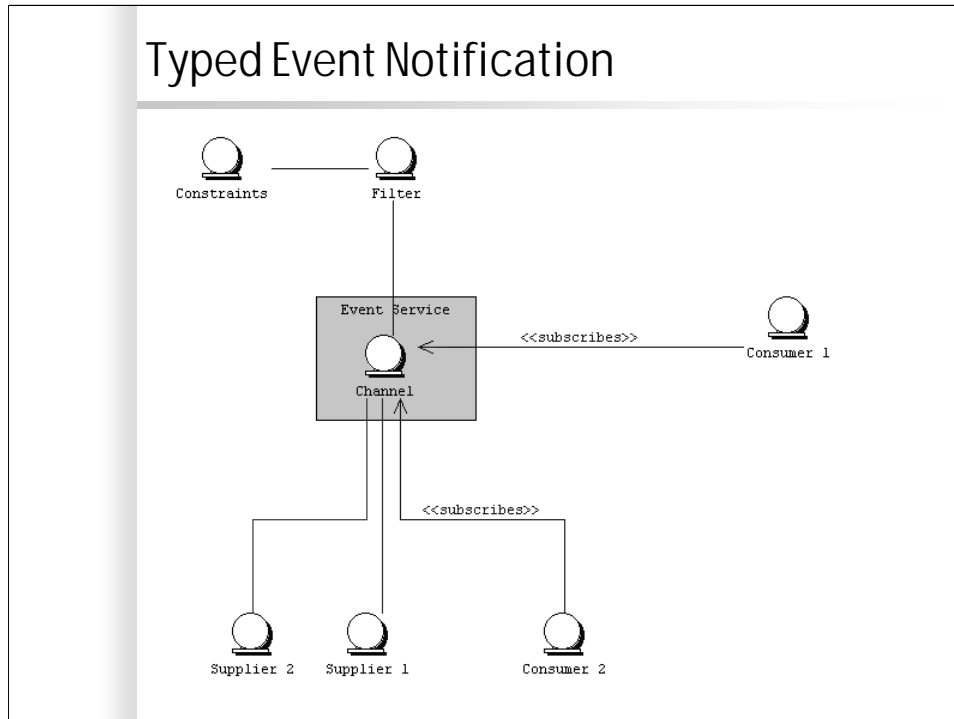
Creating “typed” events allows us to further isolate events to specific locations in the system as well as create a hierarchy of events.

This approach lays the foundation for event based exception handling in the Java runtime environment. Exceptions passed across the CORBA interface boundaries are done through typed objects rather than through standard CORBA exceptions.

## Filling in the Gaps

- ◆ *Typed events are part of the Notification specification but not the VisiBroker specification.*
- ◆ *The Notification Service needs to provide*
  - Confirmation that the event was delivered to either the event channel or the consumer of the event.

In order to fill in these gaps we developed the Event Handler subsystem.

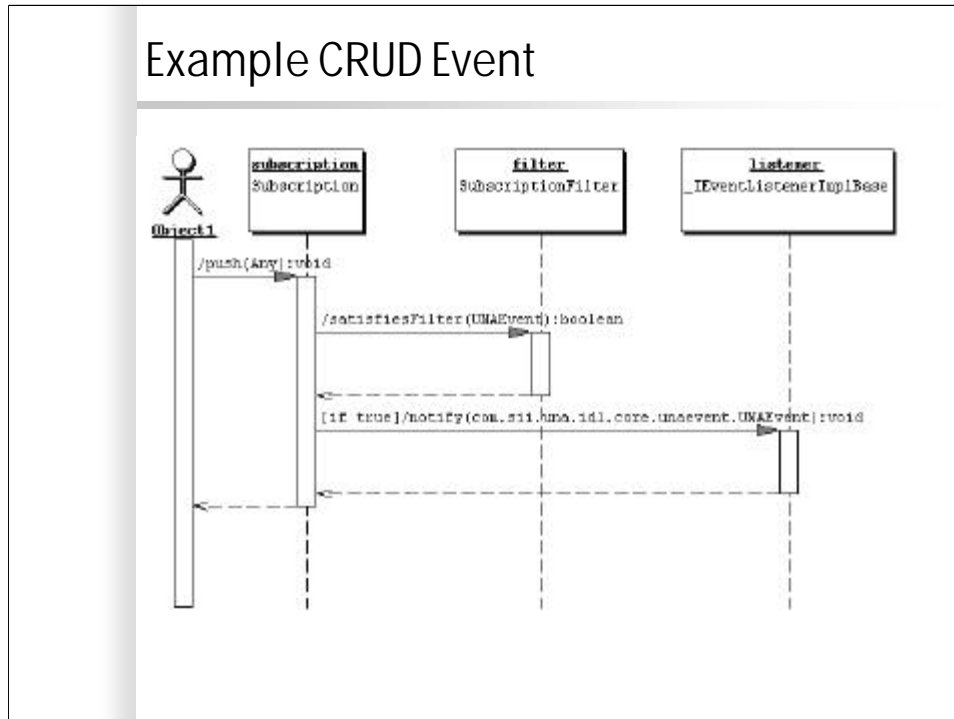


It looks like this in its simplest form.

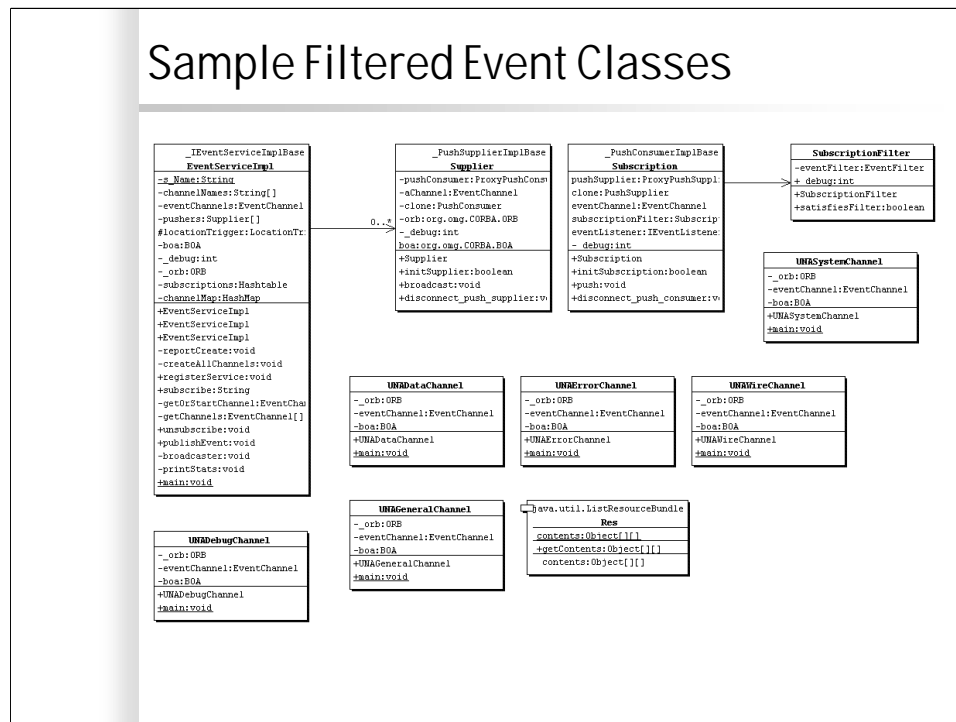
## Implementing Typed Notification

- ◆ *Typed notification is a “filtering” process*
  - Specific classes of events implemented using “Channels”
  - Consumers of events on a “Channel” apply specific filters to the messages
- ◆ *Six Channels in the system*
  - Data – CRUD events for the metadata repository
  - System – system messages
  - Error – exception messages
  - Debug – debug messages
  - General – general information exchanged between processes
  - Wire – wire handling messages

There are typed events six channels in the system. The channels may seem redundant but they provide the tools for simplifying the runtime code as well as the monitoring processes.



Here's a simple sequence process for an event.



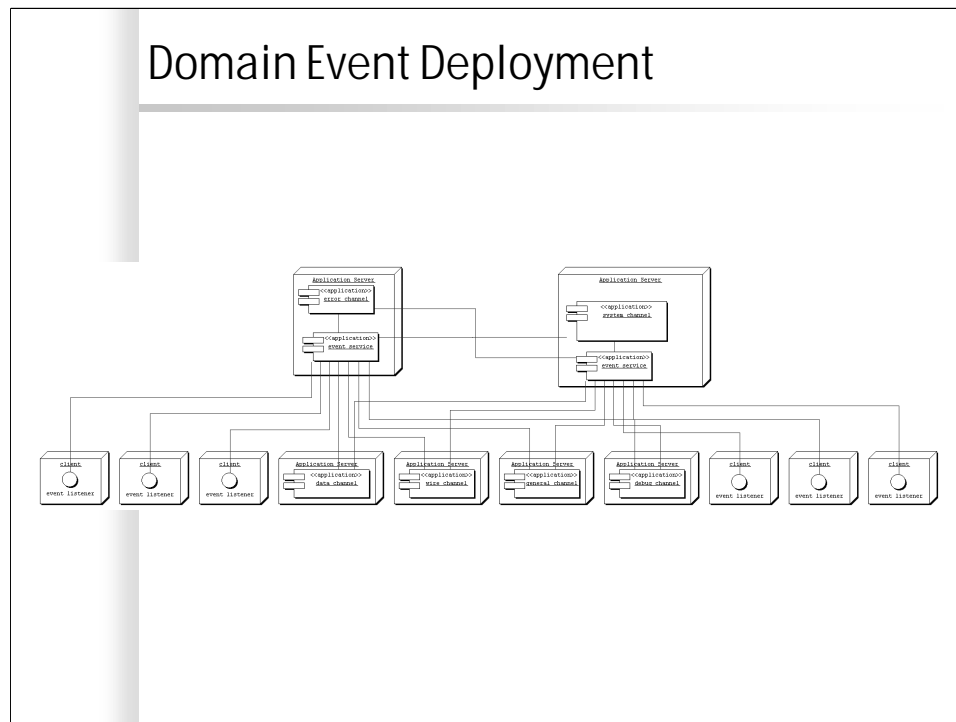
The filtering of the event takes place with the following classes.



## The Domain Event Management Concept

*Generic events are needed in the publishing system. These events must be defined, created, consumed and processes without disturbing the existing event definitions or the underlying system architecture.*

What are component based systems? Why are they different and better than traditional integrated systems? How can the user understand the differences and make an informed decision about the technology strategies needed to move into the next generation of publishing systems?



The power of components can be seen in a example of system development. The developers have built a component based infrastructure on which commercial applications can be assembled into a product. Someone (who will remain a nameless marketing executive) discovers a new requirement for a new component or subsystem that must be added to the system without disrupting the other application components. In the component environment the interfaces between the components provides the necessary isolation to allow a new application to be added with minimal impact. The business rules and software protocols shared by all components form the foundation of this process. The fundamental concept here is the market place drives this process. The marketing department will have significant input to the outcome, but the application domain and the system behaviors are controlled by external forces, not the traditional product marketing forces.



In the publishing domain, change is coming somewhat late compared to other businesses. This is an advantage that can be used to avoid many of the past difficulties.

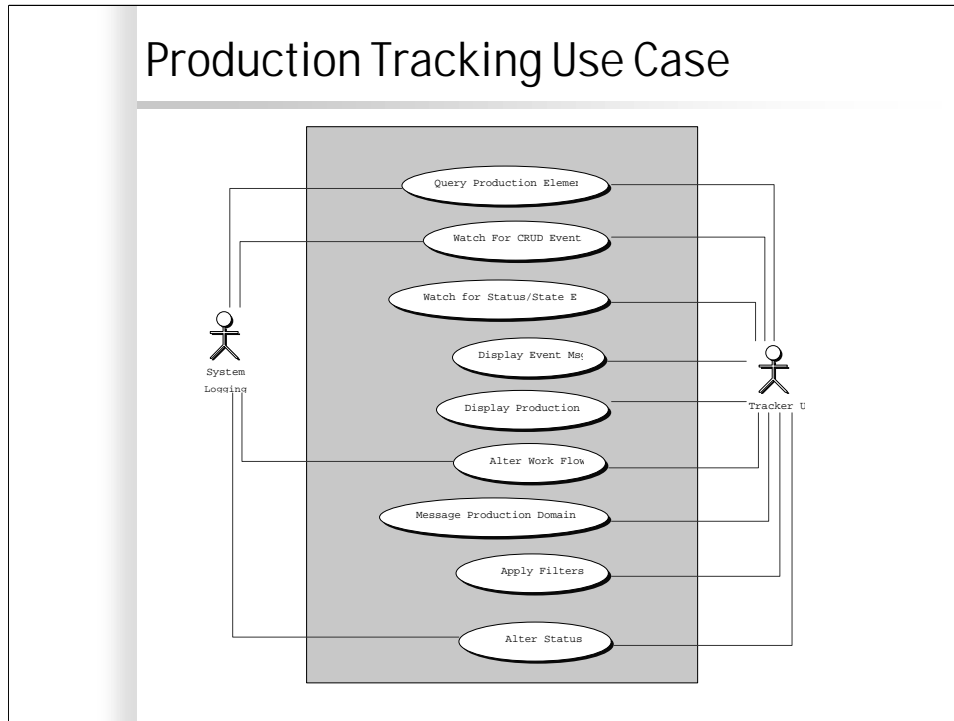
## Production Tracking Example

- ◆ *Information arrives at the publication continuously.*
- ◆ *News has an annoying habit of happening without notice.*
- ◆ *The production cycle is fairly fixed, with deadline and press start time occurring regularly throughout the day.*
- ◆ *“Production Tracking” is the means of coordinating the supply of news with the production of the paper.*

Using the events in practice is driven by the production tracking process (among others). In production tracking asynchronous events concur throughout the system. They are coordinated by the page editor as well as the production planning staff.

The makeup of the newspaper and the pages is complex because the source of the printable material is both changing and fixed. News changes all the time, advertising is booked months in advance. Ads have specific positions on the page, while news has various “lengths” and must be fit around the ads. Ads sell papers, news is the purpose of journalism.

Coordinating these two forces is tricky and on “deadline” the layout of the paper becomes critical.



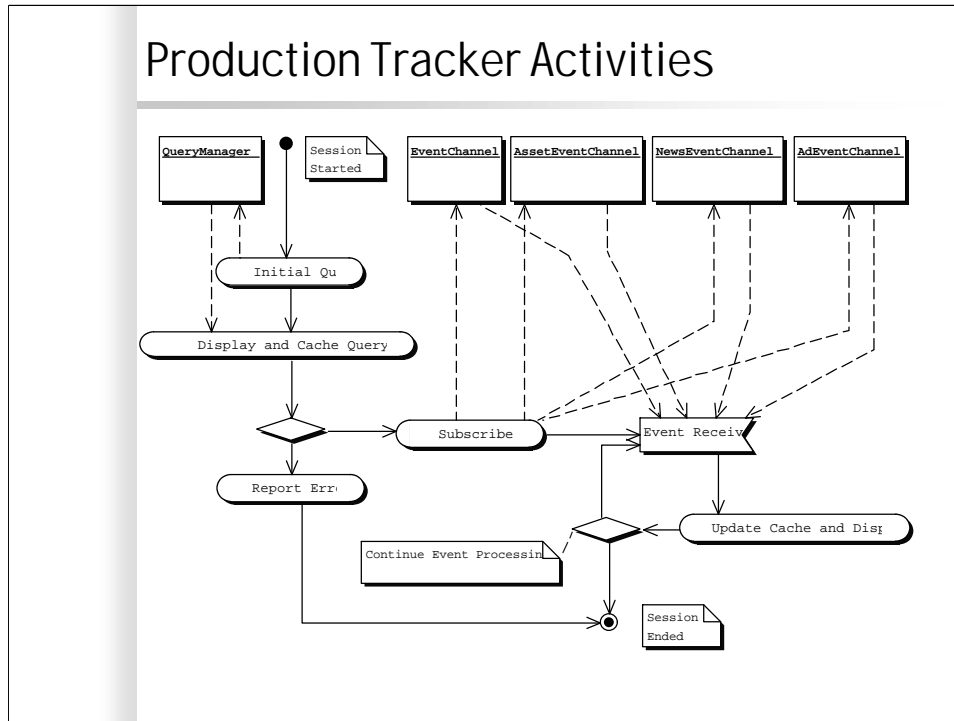
These are the typical participants and the process in the production tracking Use Case

## Production Tracking Use Case

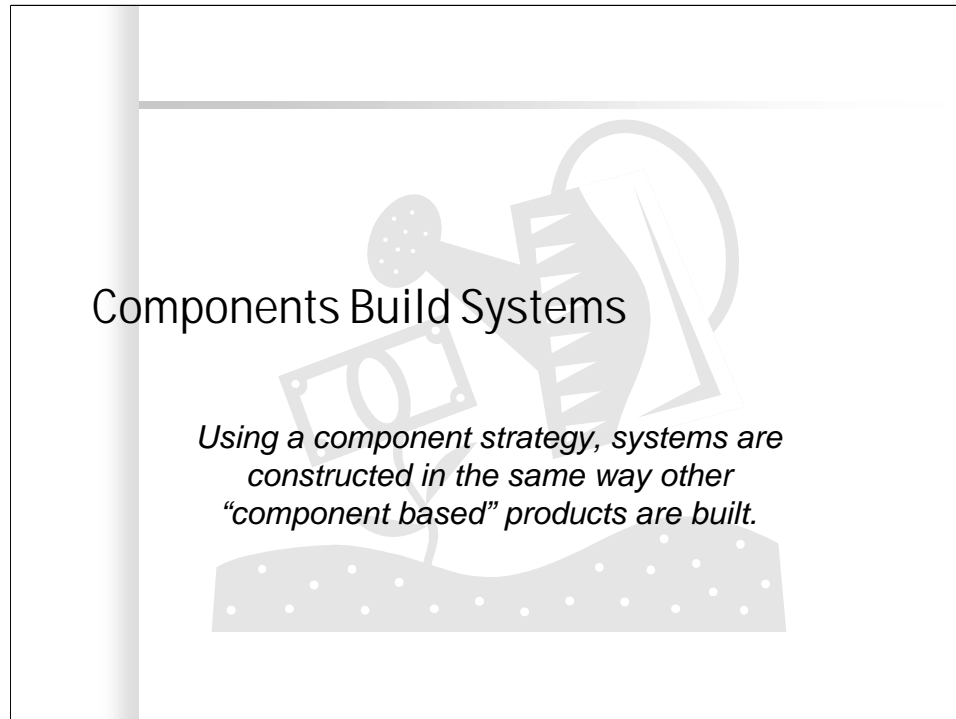
- ◆ *Issue a query against the metadata repository*
  - Returns a list of “pages” to be tracked
  - Indicate which pages are of interest to the user
- ◆ *“Watch” these pages for CRUD events*
  - New items added to page
  - Items deleted from page
  - New pages inserted before or after this page
- ◆ *Changes to the “watched” page are important for the Page Layout system*
  - Pages can only be printed in groups of four (hold your newspaper up and look at the actual ink on paper).
  - “Page Budgets” are important to advertising as well as editorial.

The coordination of the tracking process is based on “change”. To detect change a query is needed to each domain that can hold the content that may have changed. This is usually called a “watch” process.

The synchronization of the text, photos, art work, layout and pagination can all occur in parallel. So the “watch” processes themselves must be coordinated.



Here's a simplified view of the participants and their processes



Once a component strategy has been selected, the assembly of these components into a system is driven by the business architecture not the technical architecture.



The middleware market is an enabling technology market. This is different than an integration technology using API's, or a technology that allows items to be connected (RMI for example). This software is independent of the applications and at the same time is the glue that holds them together.