

Section 7.2 –
Architectural Design

Software
Engineering

Software Architectures

SW architecture community has codified architecture knowledge in

- Architectural styles or patterns
- Architectural tactics
- Reference architectures

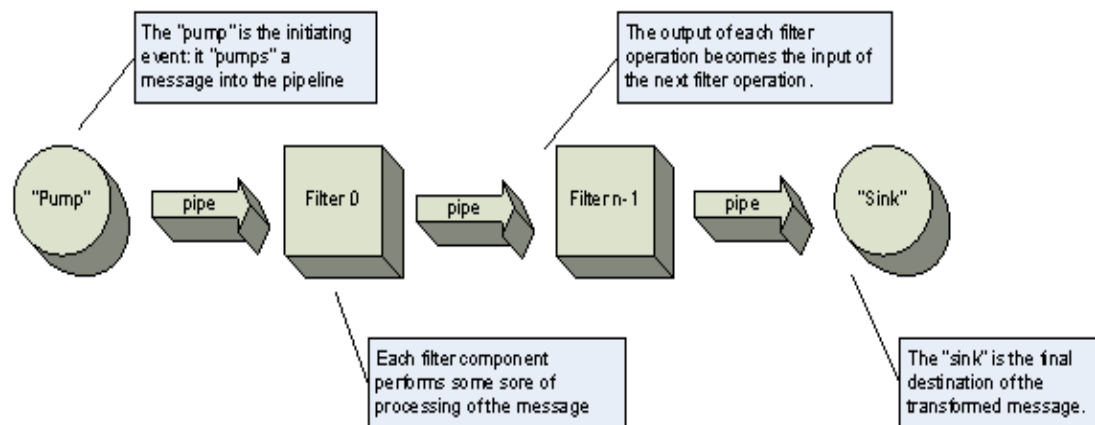
Pipes and filters

The **filter** transforms or filters the data it receives via the pipes with which it is connected. A filter can have any number of input pipes and any number of output pipes.

The **pipe** is the connector that passes data from one filter to the next. It is a directional stream of data, that is usually implemented by a data buffer to store all data, until the next filter has time to process it.

The **pump** or producer is the data source. It can be a static text file, or a keyboard input device, continuously creating new data.

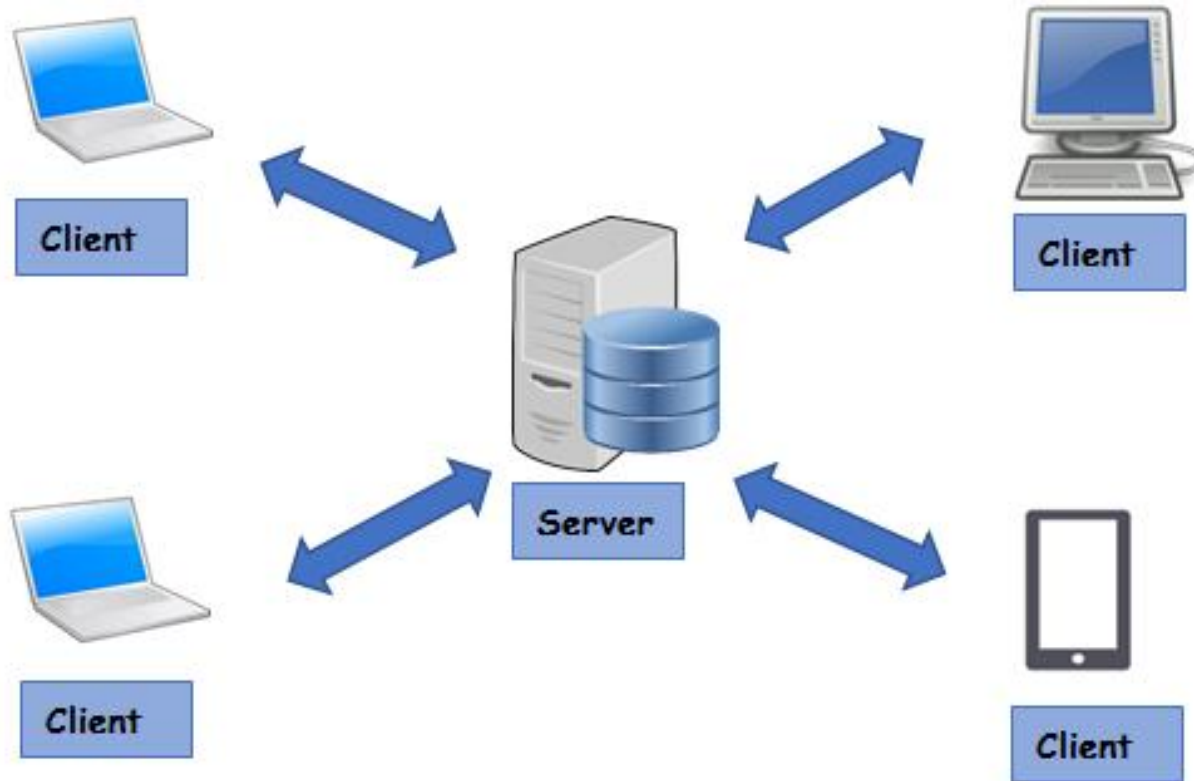
The **sink** or consumer is the data target. It can be another file, a database, or a computer screen.



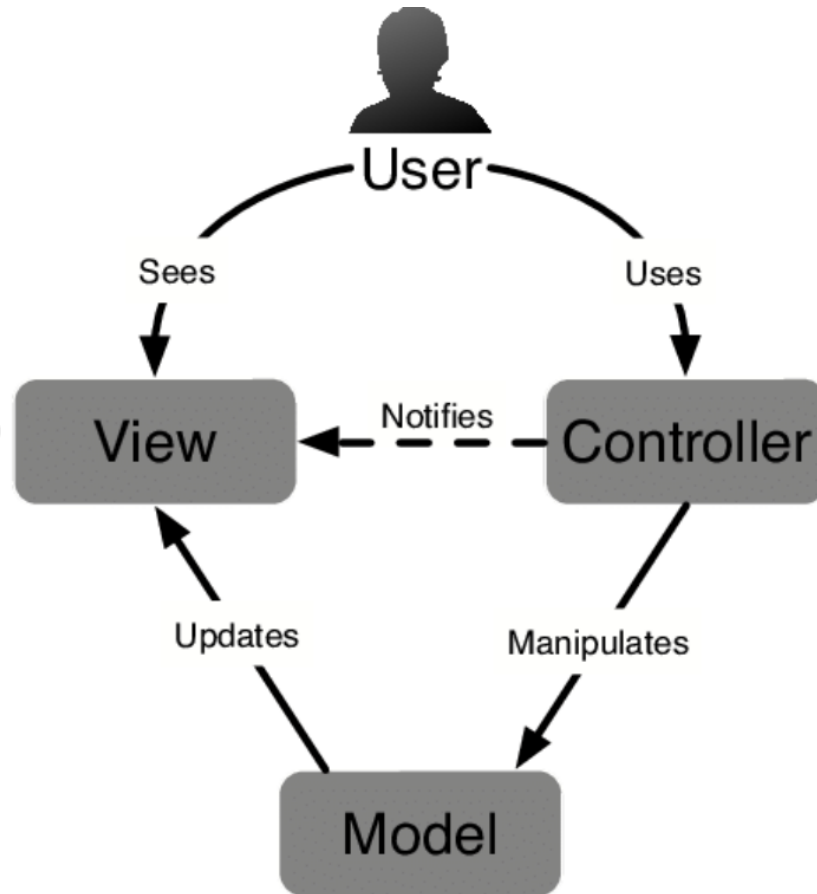
Event driven



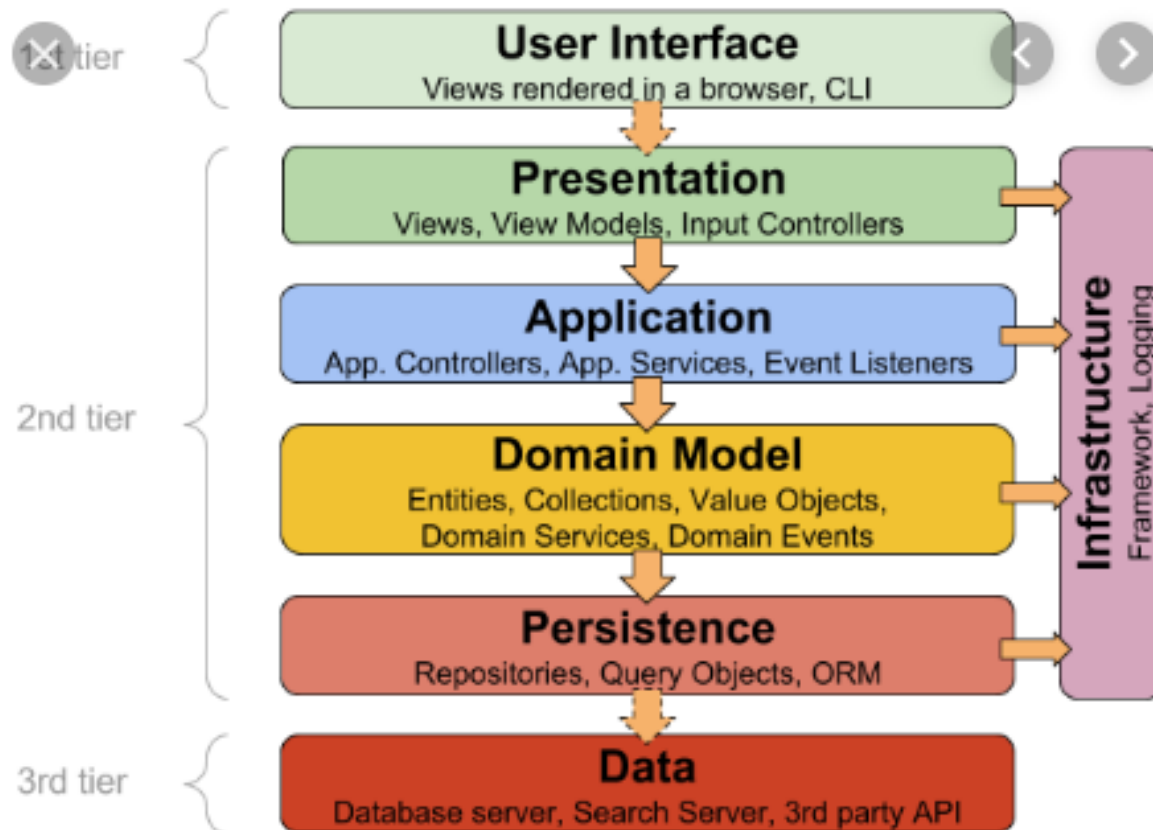
Client-Server



Model-View-Controller



Layered



Database-centric

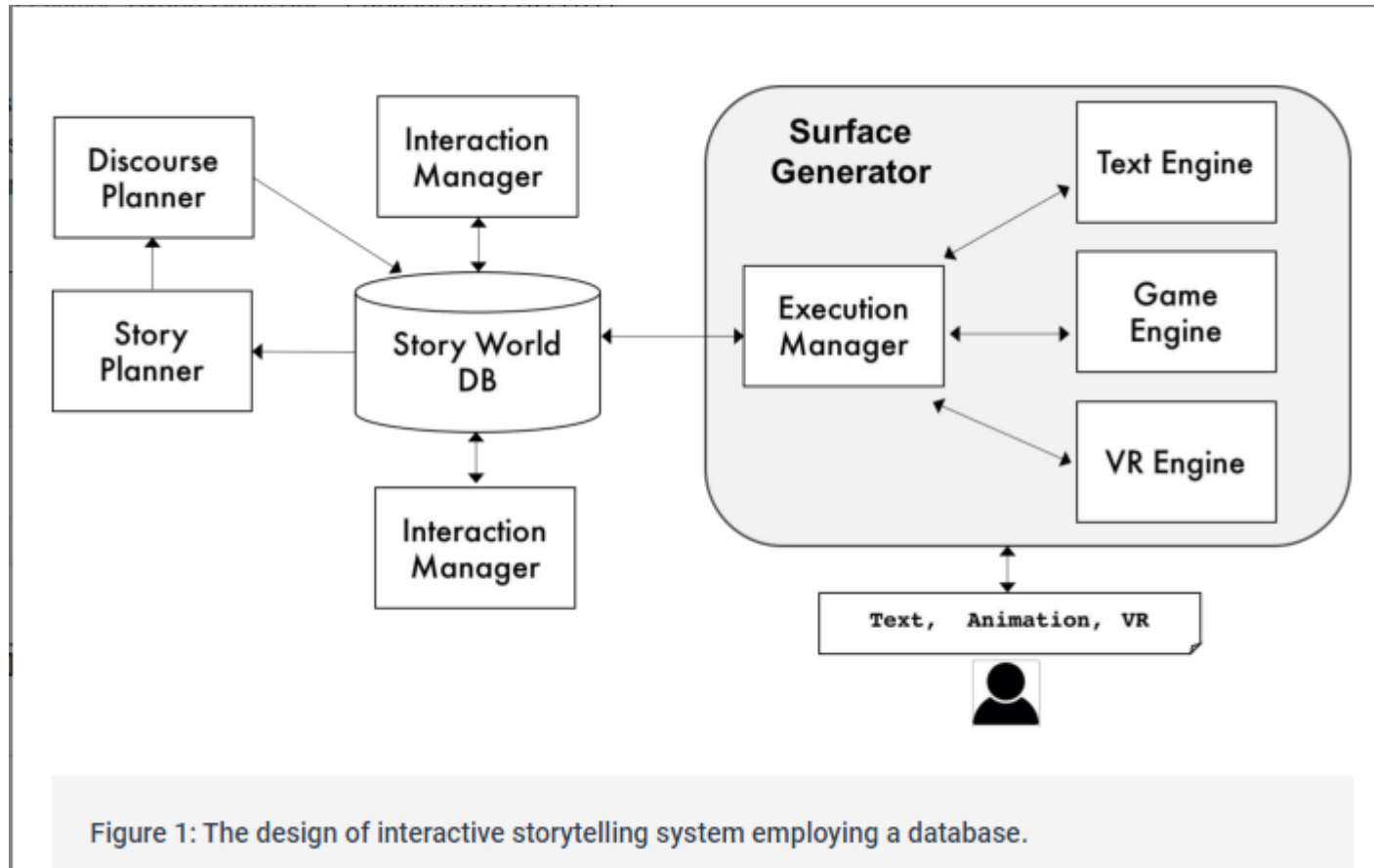
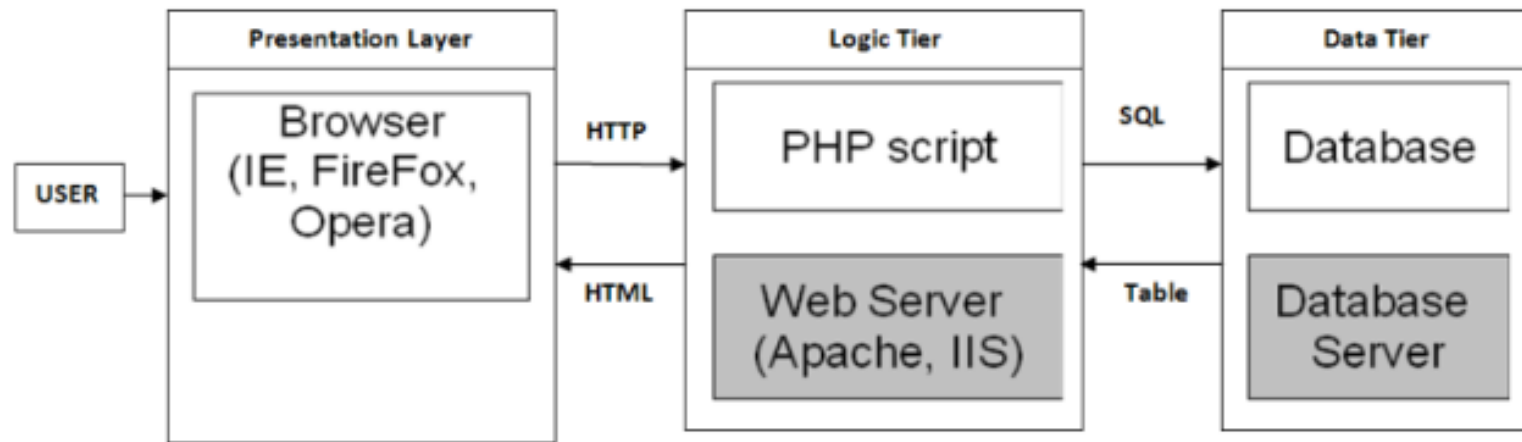


Figure 1: The design of interactive storytelling system employing a database.

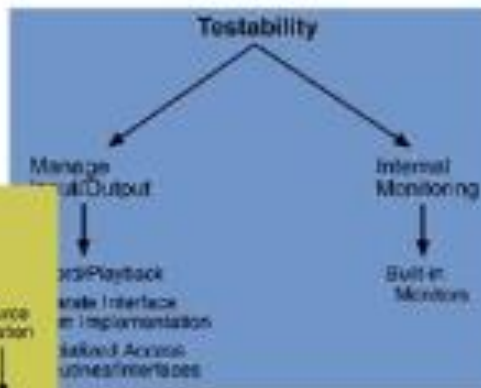
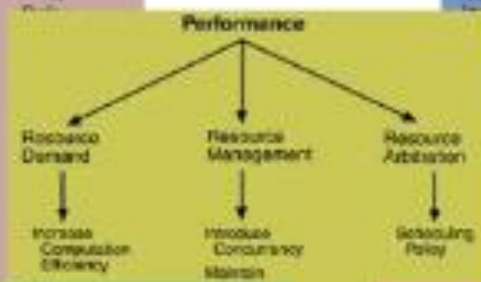
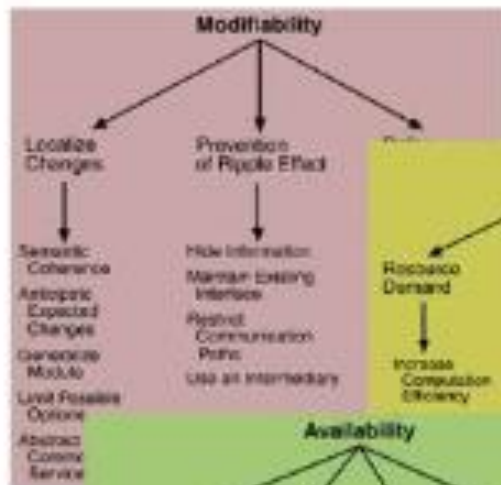
Three-tier

Three-tier Architecture Example

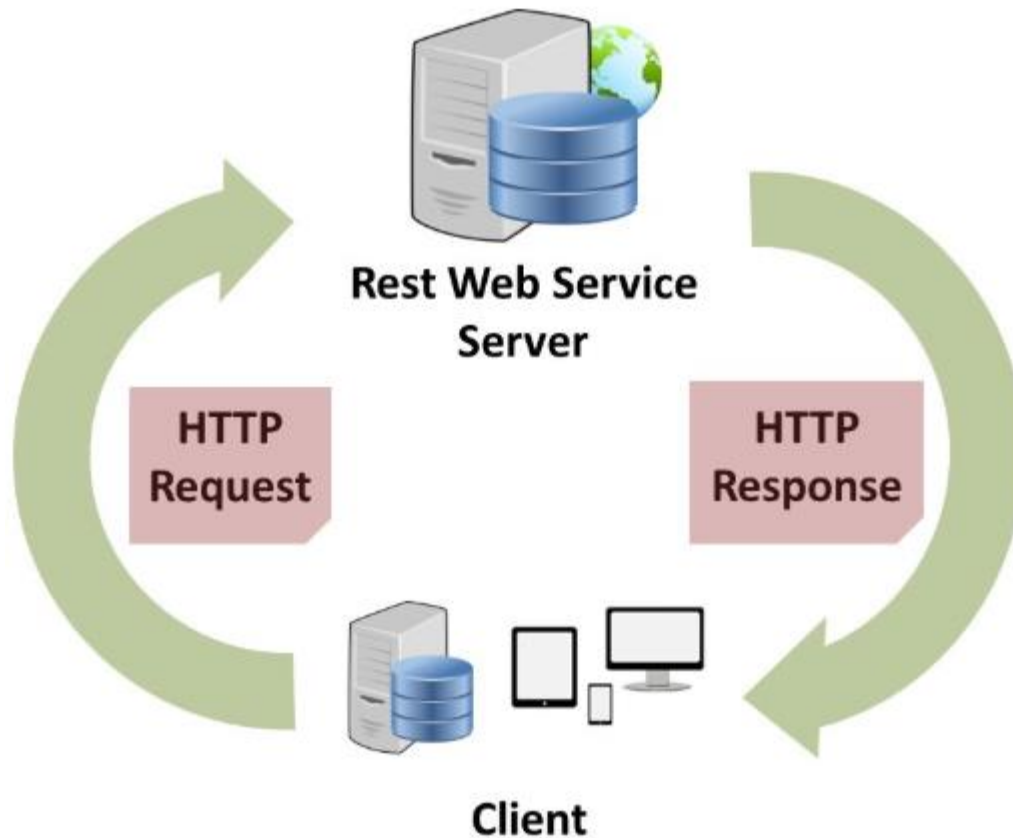


Architectural Tactics

Architectural Tactics



Reference Architecture - REST



<https://medium.com/@sagar.mane006/understanding-rest-representational-state-transfer-85256b9424aa>

Representational State Transfer, REST Architectural Style

Well-designed Web application

- Network of web pages (a virtual state-machine)
- User progresses through an application by selecting links (state transitions)
- Resulting in the next page (representing the next state of the application)

REST Architectural Style

REST style has formal and informal “constraints:

- Client-server
- Stateless
- Cacheable
- Uniform interface
- Layered system
- Code on demand (optional)

<https://medium.com/@sagar.mane006/understanding-rest-representational-state-transfer-85256b9424aa>

REST Architectural Style

Steps

- Define the domain and data.
- Organize the data into groups.
- Create URI to resource mapping.
- Define the representations to the client (XML, HTML, CSS, ...).
- Link data across resources (connectedness or hypermedia).
- Create use cases to map events/usage.
- Plan for things going wrong.