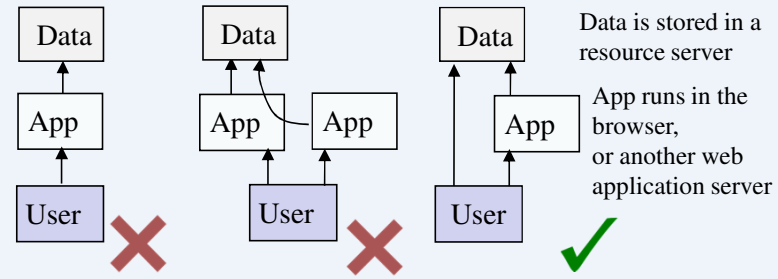
# Building communicating web applications leveraging endpoints and cloud resource service

Cloud Computing

## Problem

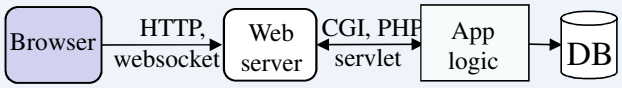
Existing social apps suffer from these problems:

1. Redundancy
2. Application lock-in
3. Rigid data boundary
4. Tied lifetime of data

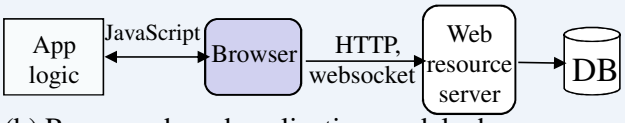



(a) app controls user data (b) app allows another to access (c) user controls her data and approves apps

## Resource Model




(a) Traditional web application model

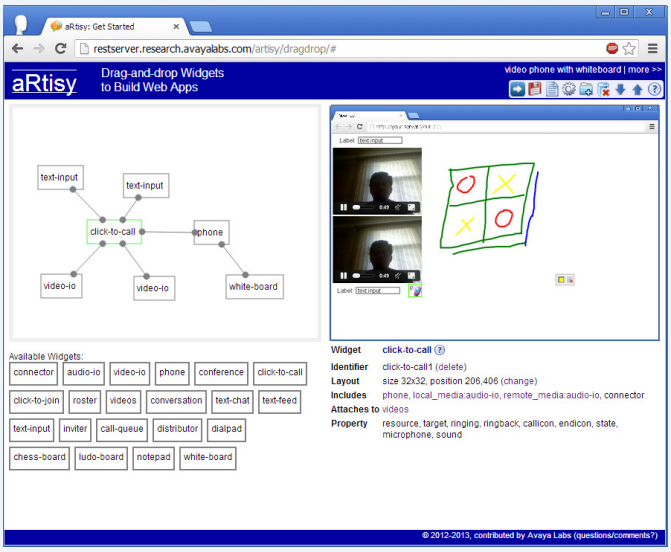



(b) Resource-based application model where app-logic runs in the client browser in JavaScript

1. Resource server is independent of web application, e.g., public web-site can bind to private database.
2. Hierarchical file system style resources, with JSON for entity representation.
3. Promotes model-view style application development, with resource level app mash-ups.

## aRtisy App Builder and Widgets

An example app development in progress for two-party video phone with shared white-board.




1. What are the various communication widgets and how do they interact and mash-up at the resource level?
2. How does it provide a generic signaling path for WebRTC – emerging web real-time communications?
3. What various applications from video chat, video presence, social wall did we build using this platform?
4. What are the security, robustness and interoperability challenges?


## Summary:

aRtisy is a developer platform and SDK to create and host communicating applications.

The application logic is in client side JavaScript and the server is just a data store with interface to access data and events.

The resource server can be separate from the website, in an on-premise or cloud, or a hybrid deployment.

aRtisy Widgets are small, light-weight and implement ready-made communication scenarios in HTML/JS/CSS.


AVAYA labs


Kundan Singh <singh173@avaya.com>
Venkatesh Krishnaswamy <venky@avaya.com>