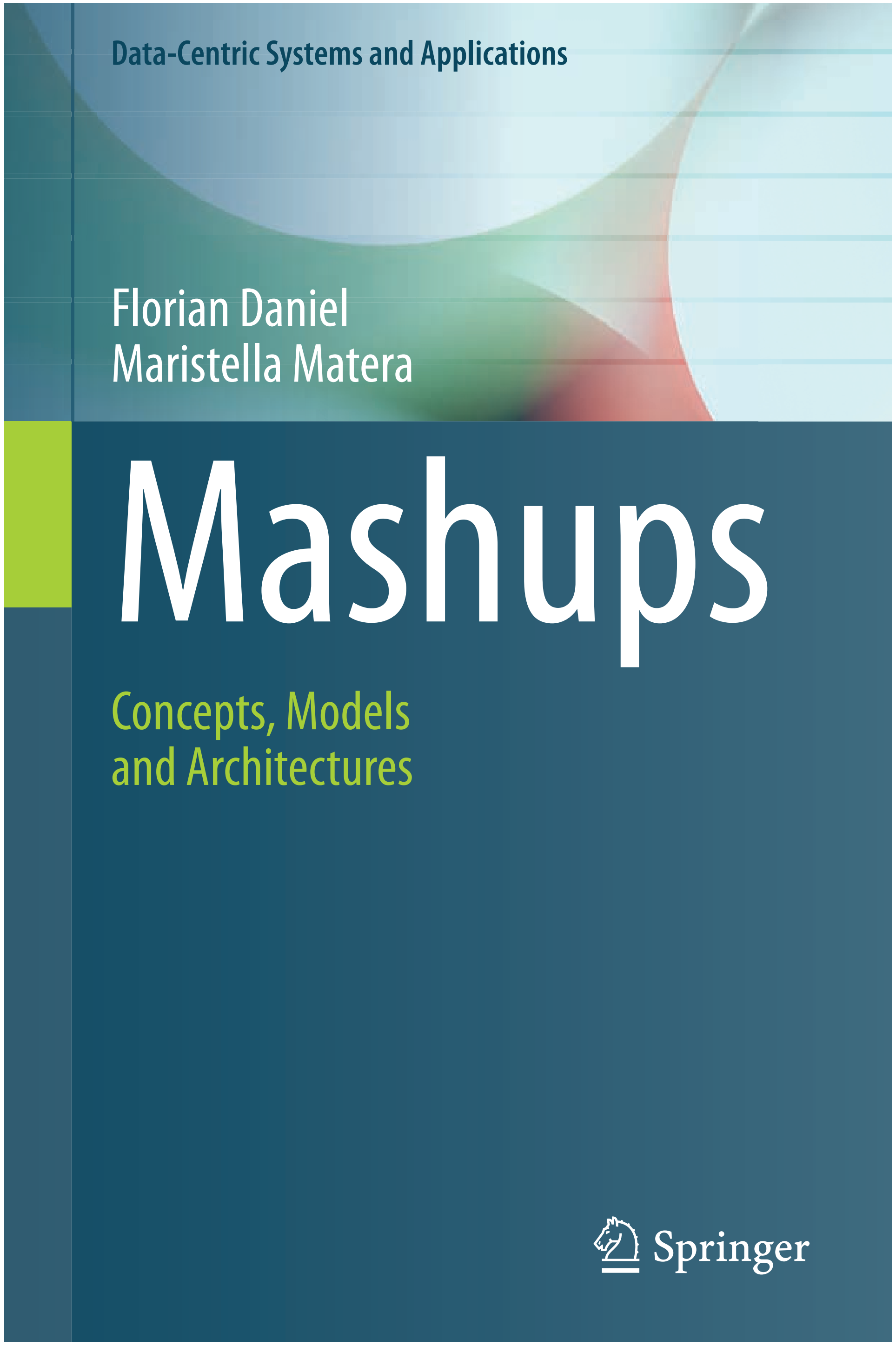

Chapter 6
Mashups

Figures

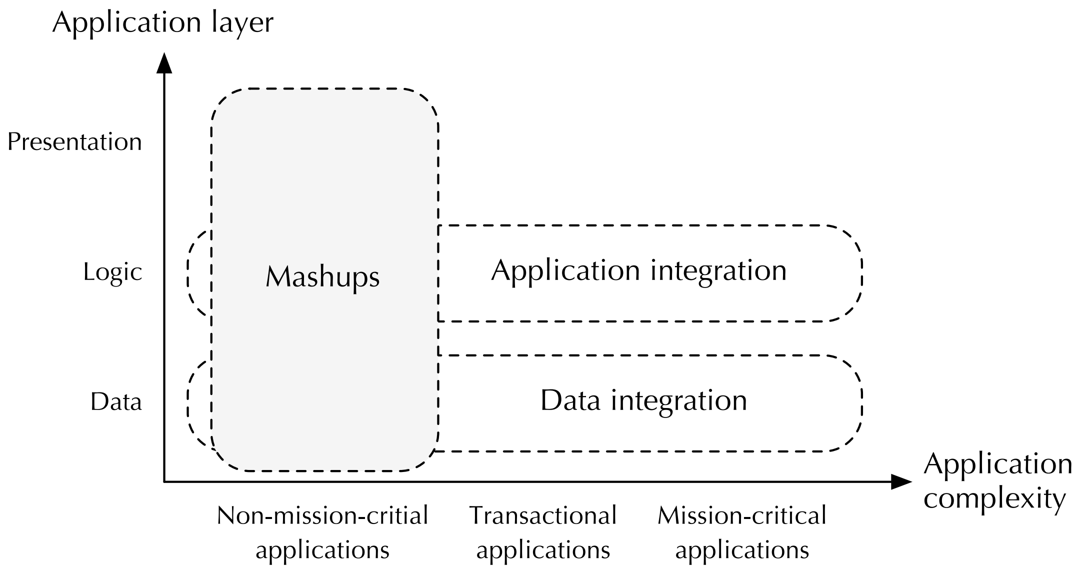

Fig. 6.1 Positioning of mashups compared to other integration practices, such as application integration and data integration. Mashups introduce integration at the presentation layer and typically focus on non-mission-critical applications.

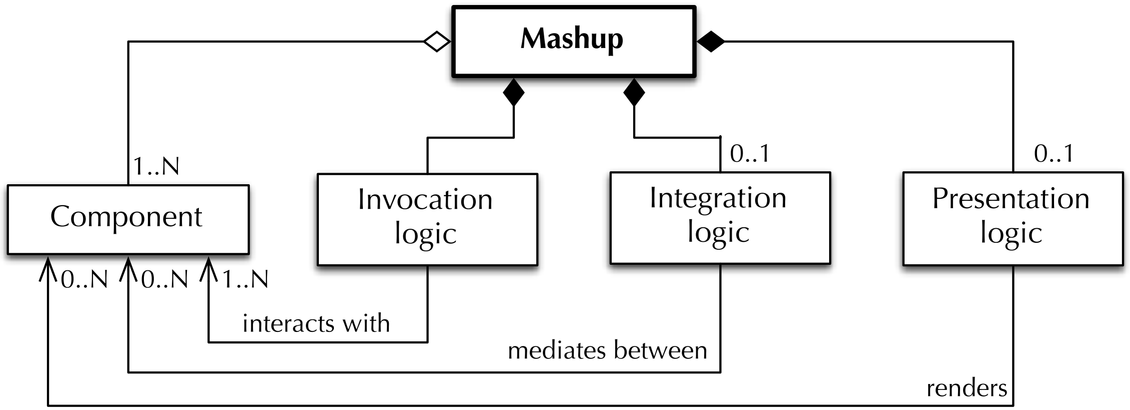

Fig. 6.2 The basic mashup model: a mashup integrates a set of components, possibly puts them into communication, and optionally renders results or components.

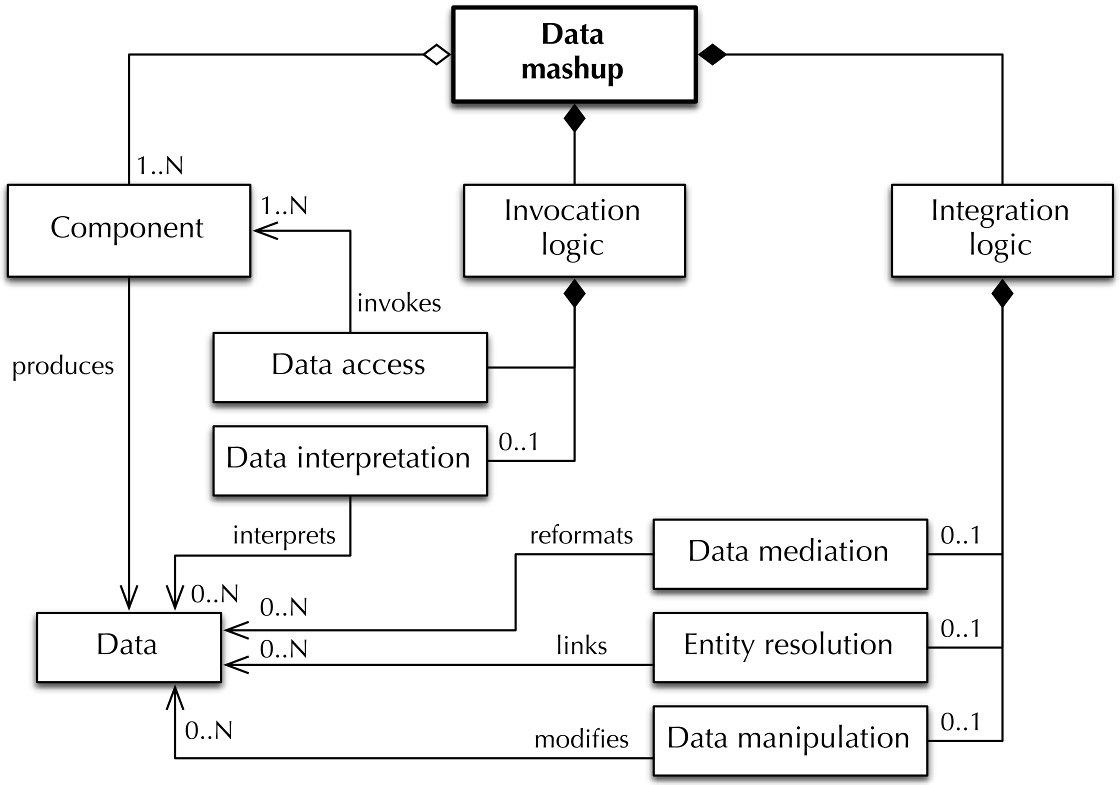

Fig. 6.3 A conceptual model for data mashups.

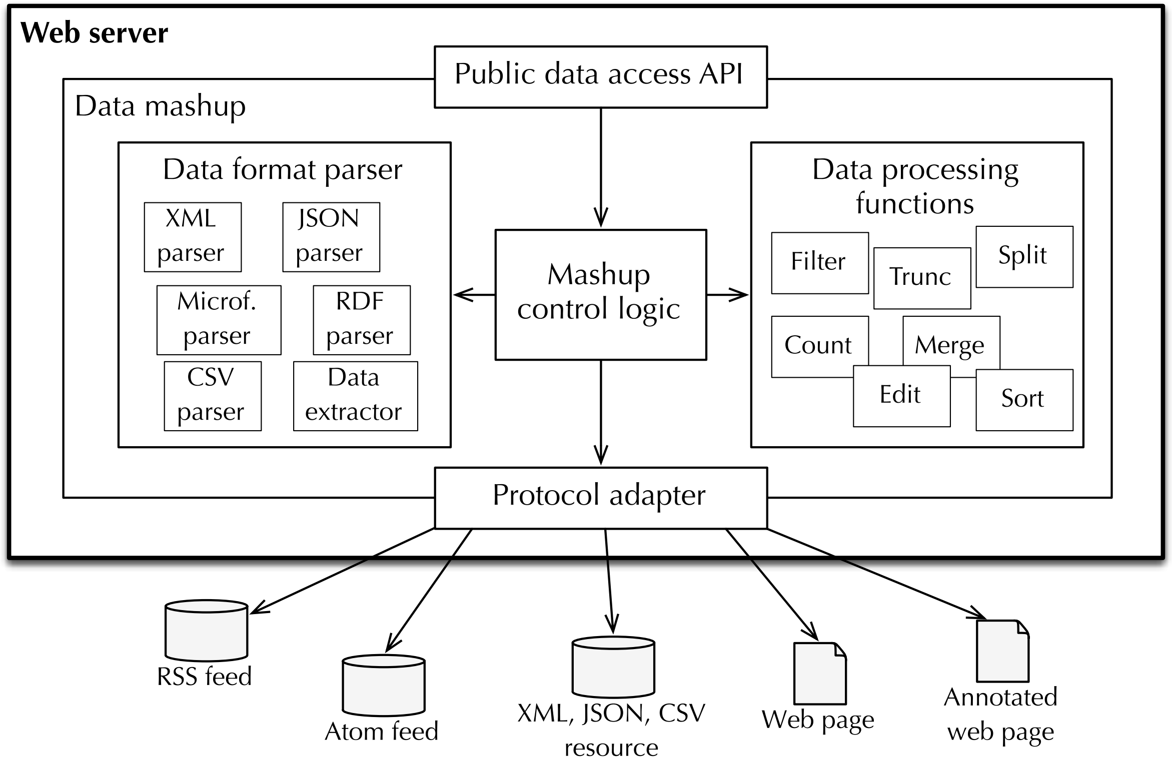

Fig. 6.4 Basic data mashup architecture with direct data passing among data processing functions.

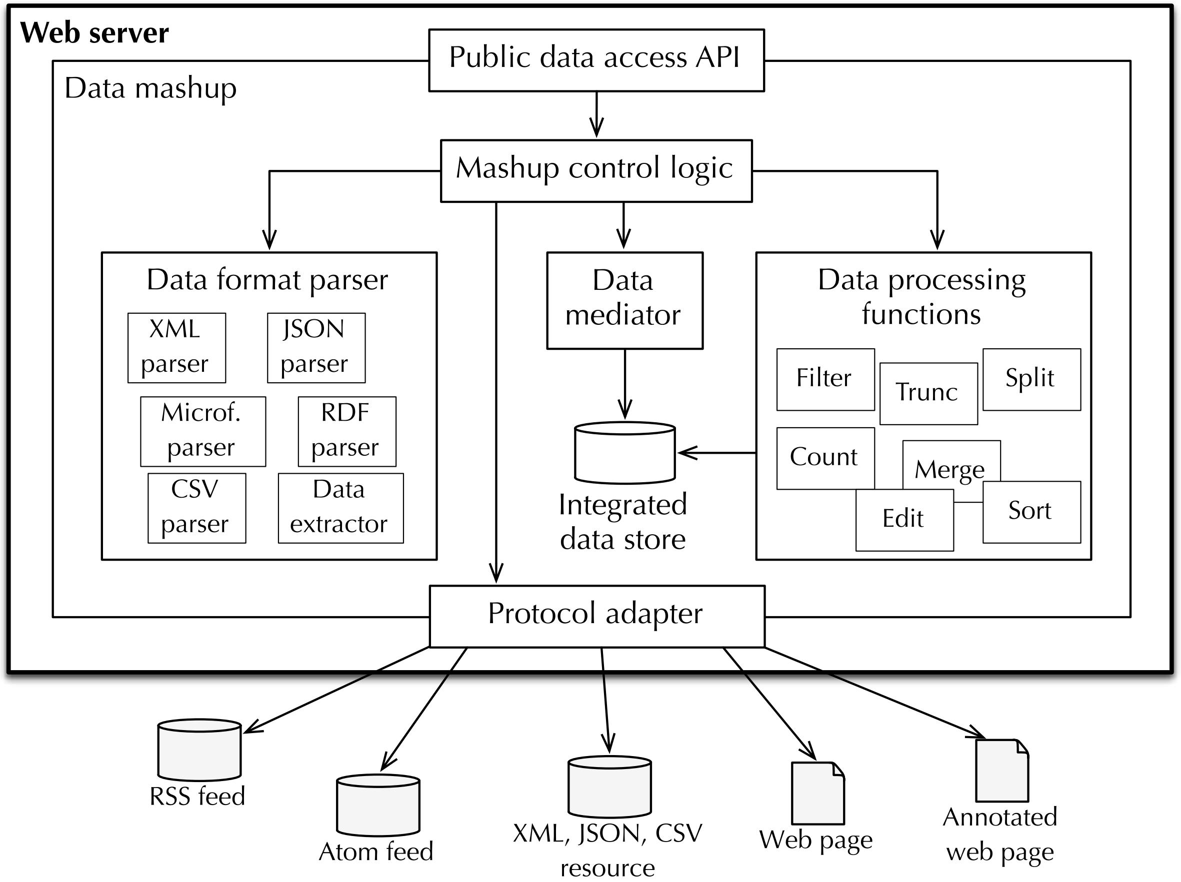

Fig. 6.5 Data mashup architecture with data mediation and integrated data store.

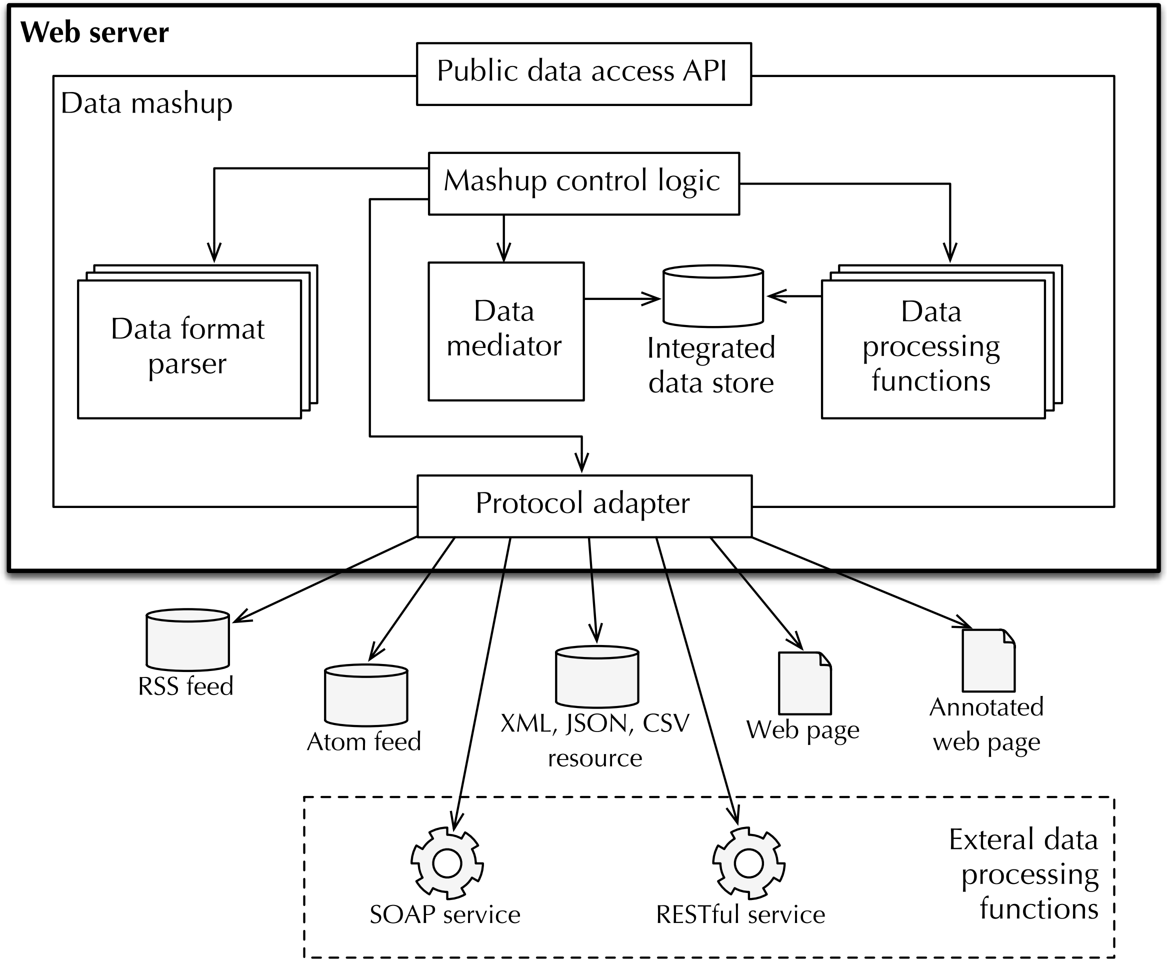

Fig. 6.6 Architecture of a data mashup with external data processing logic.

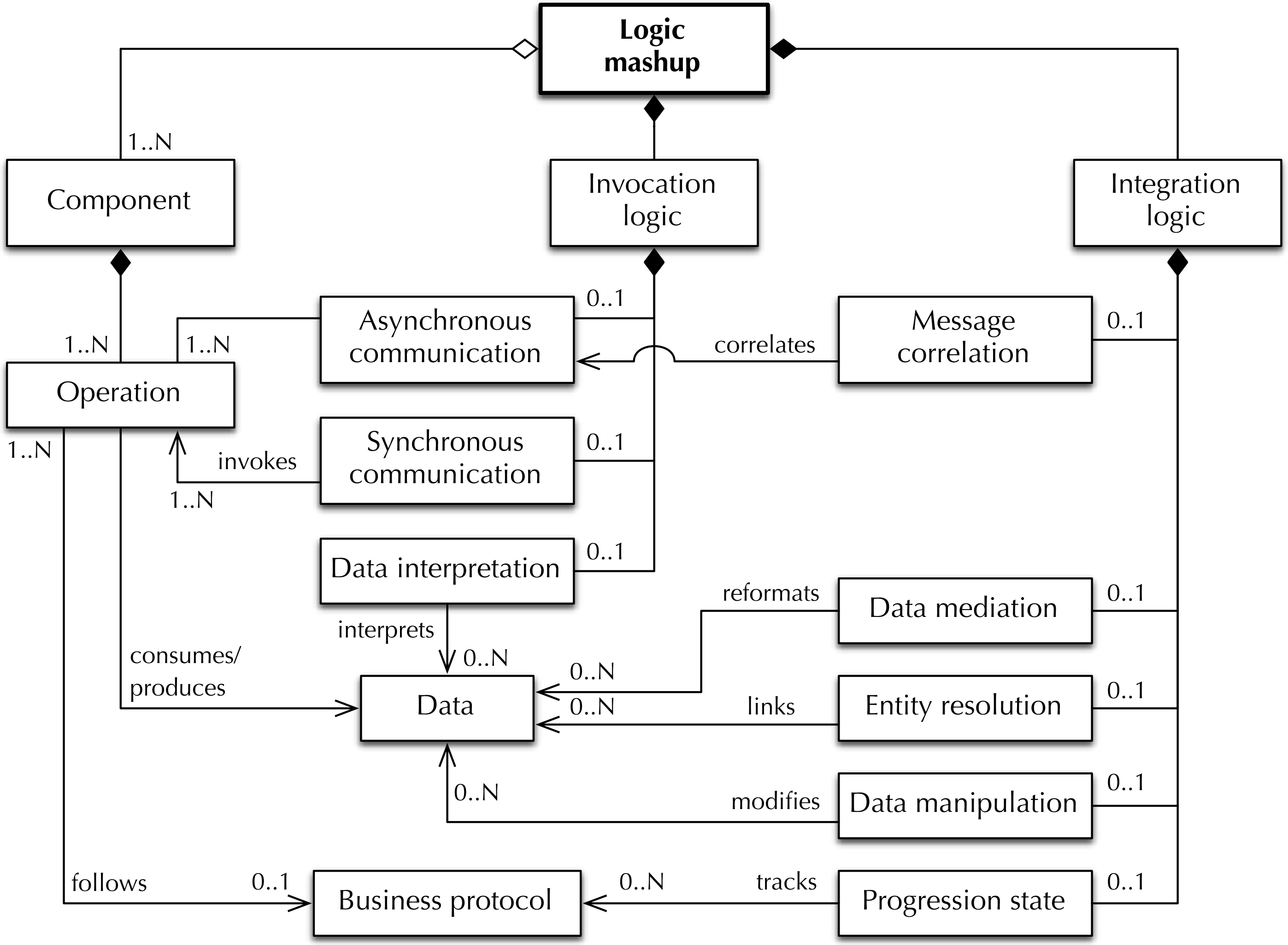

Fig. 6.7 Mashup model with support for stateless and long-living logic mashups.

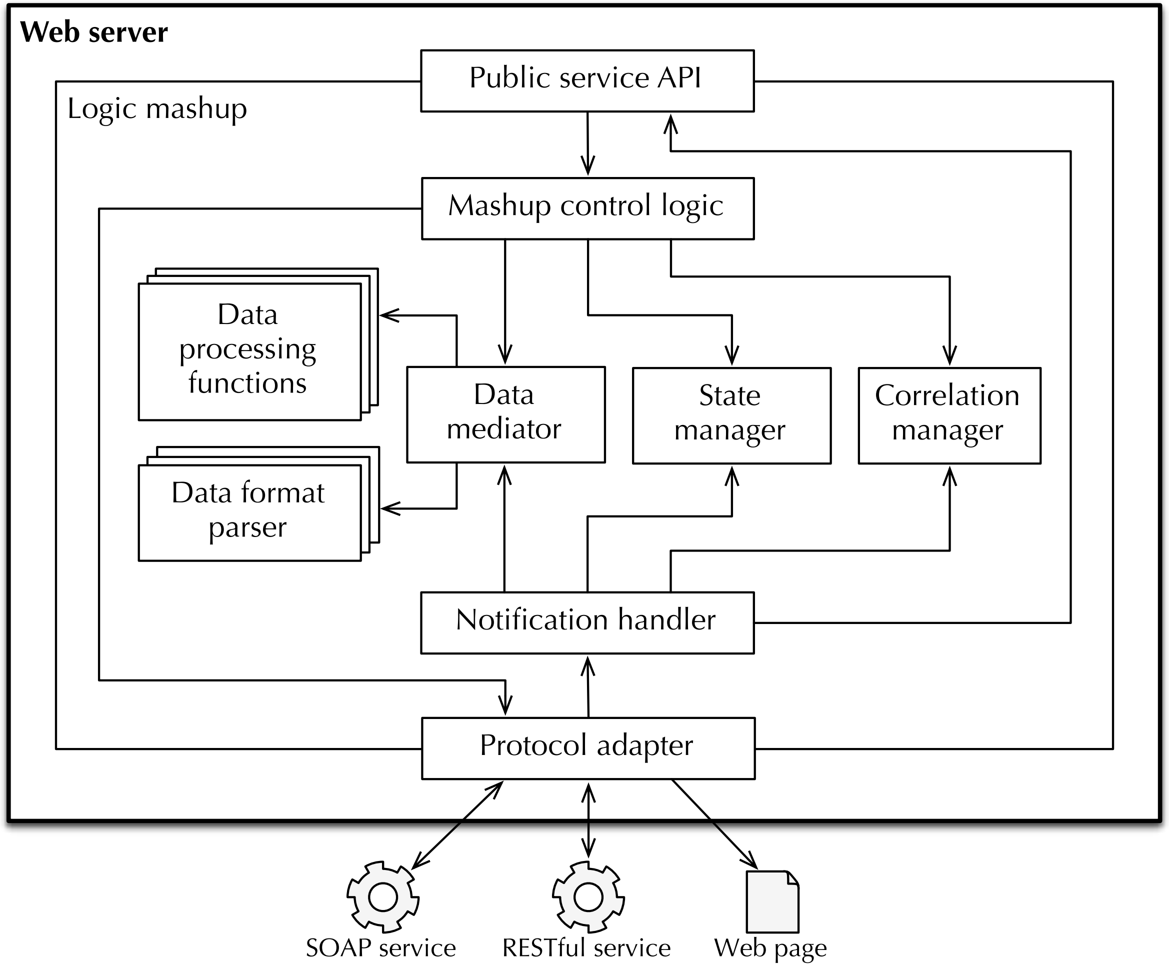

Fig. 6.8 Generic architecture of logic mashups with support for synchronous and asynchronous communication; data are processed like for data mashups.

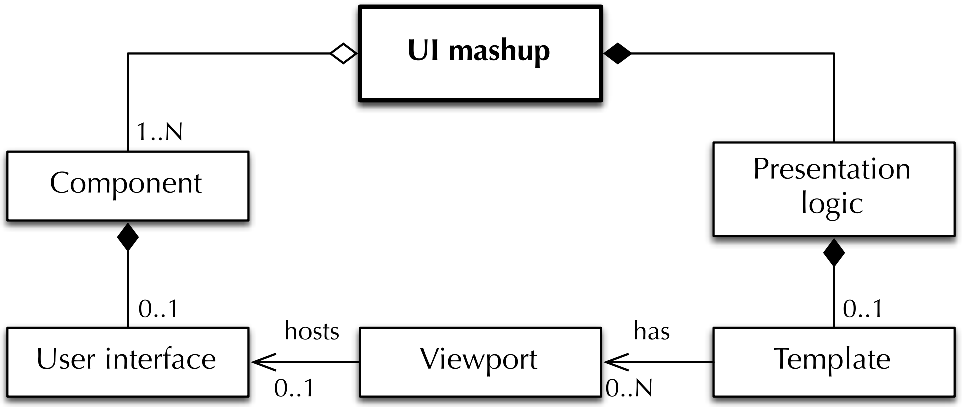

Fig. 6.9 User interface mashup model without inter-component communication.

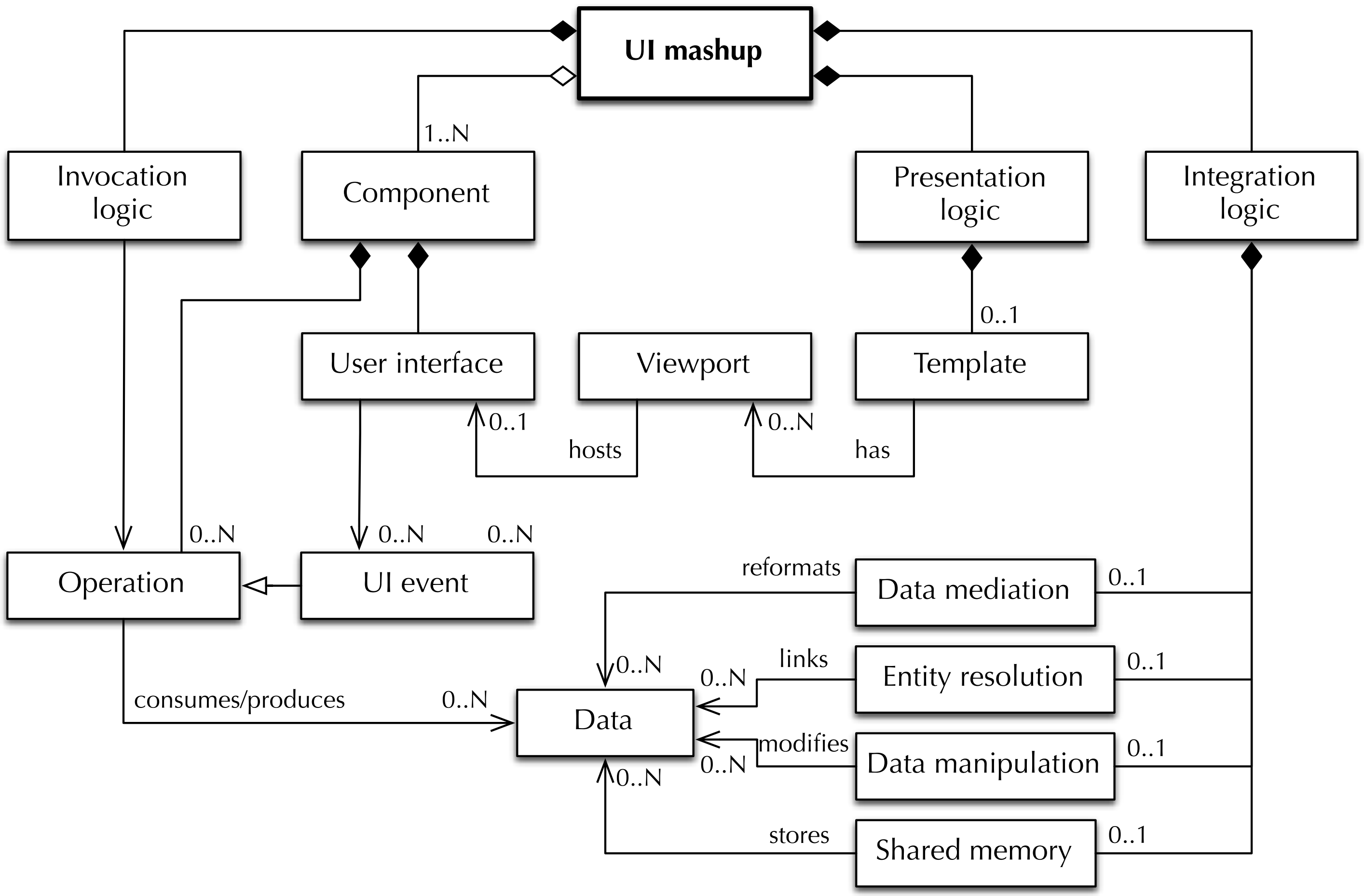

Fig. 6.10 User interface mashup model with inter-component communication.

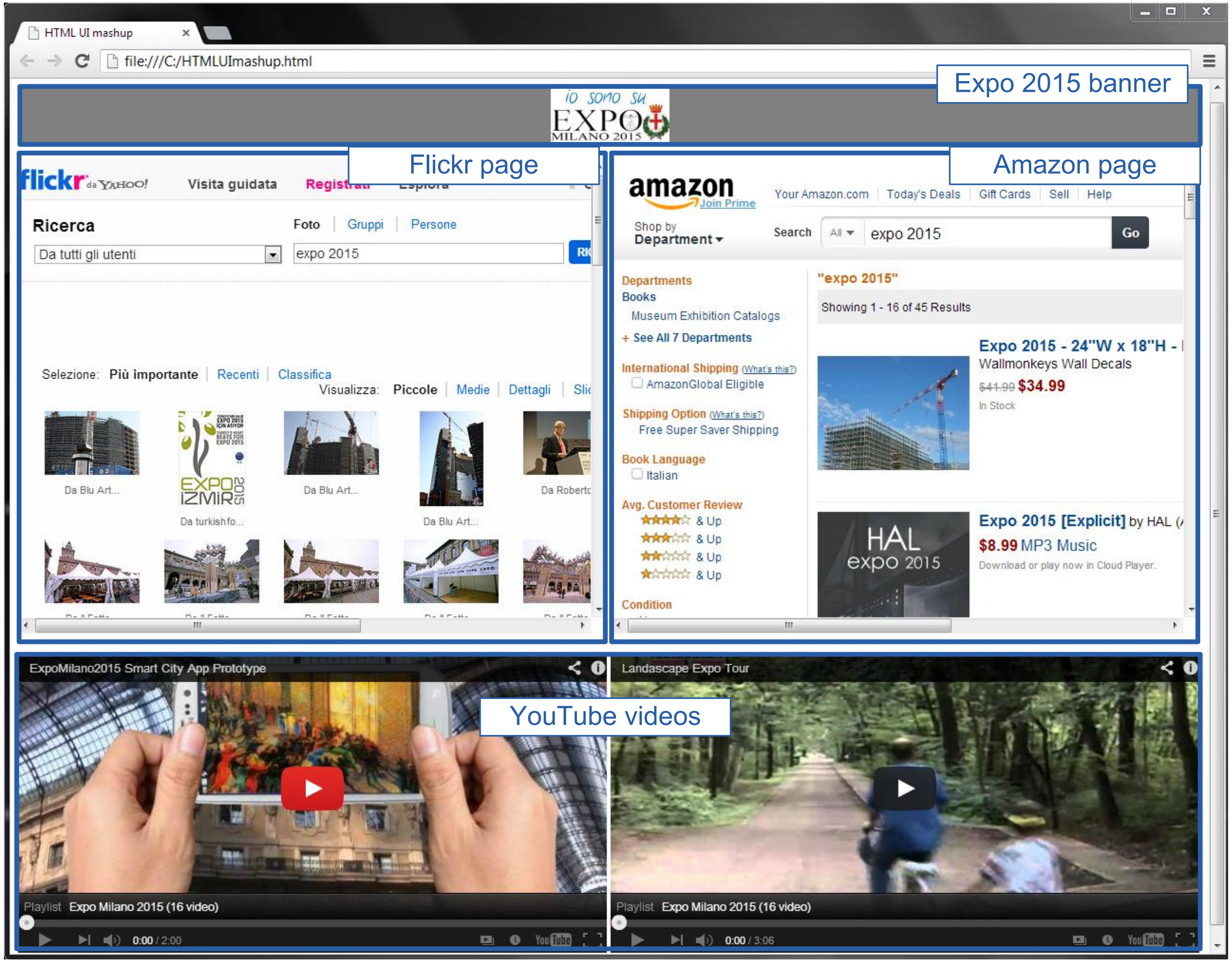

Fig. 6.11 The simplest UI mashup: embedding external resources inside own HTML code.

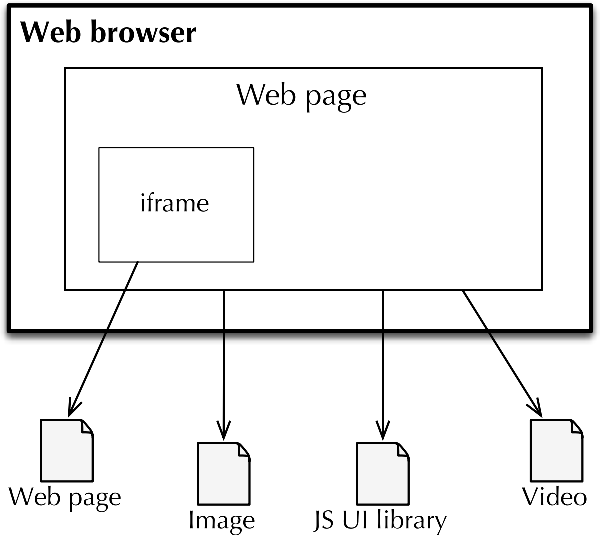

Fig. 6.12 Architecture for a simple HTML mashup. It embeds, within a Web page, HTML snippets for displaying contents retrieved in a remote site and JavaScript code to interact with a remote JS library or API.

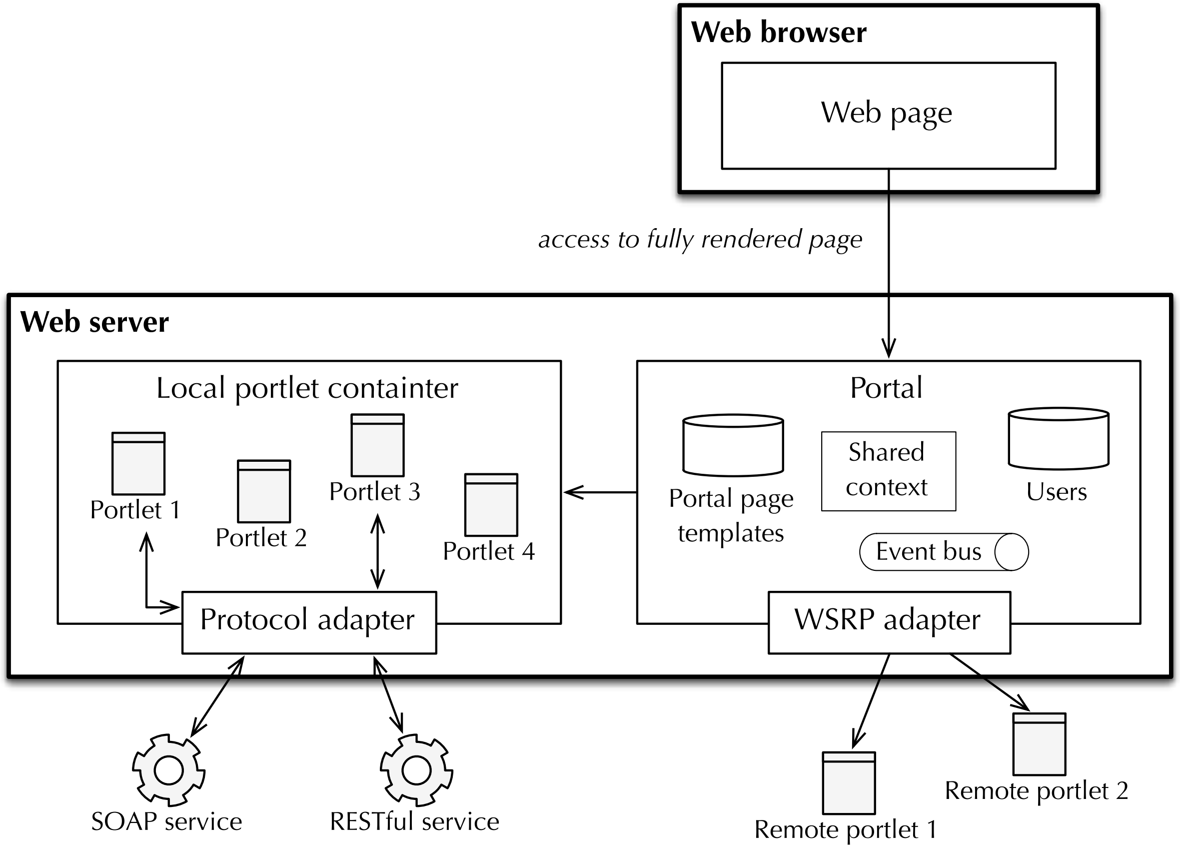

Fig. 6.13 Simplified architecture of a portal serving local and remote portlets.

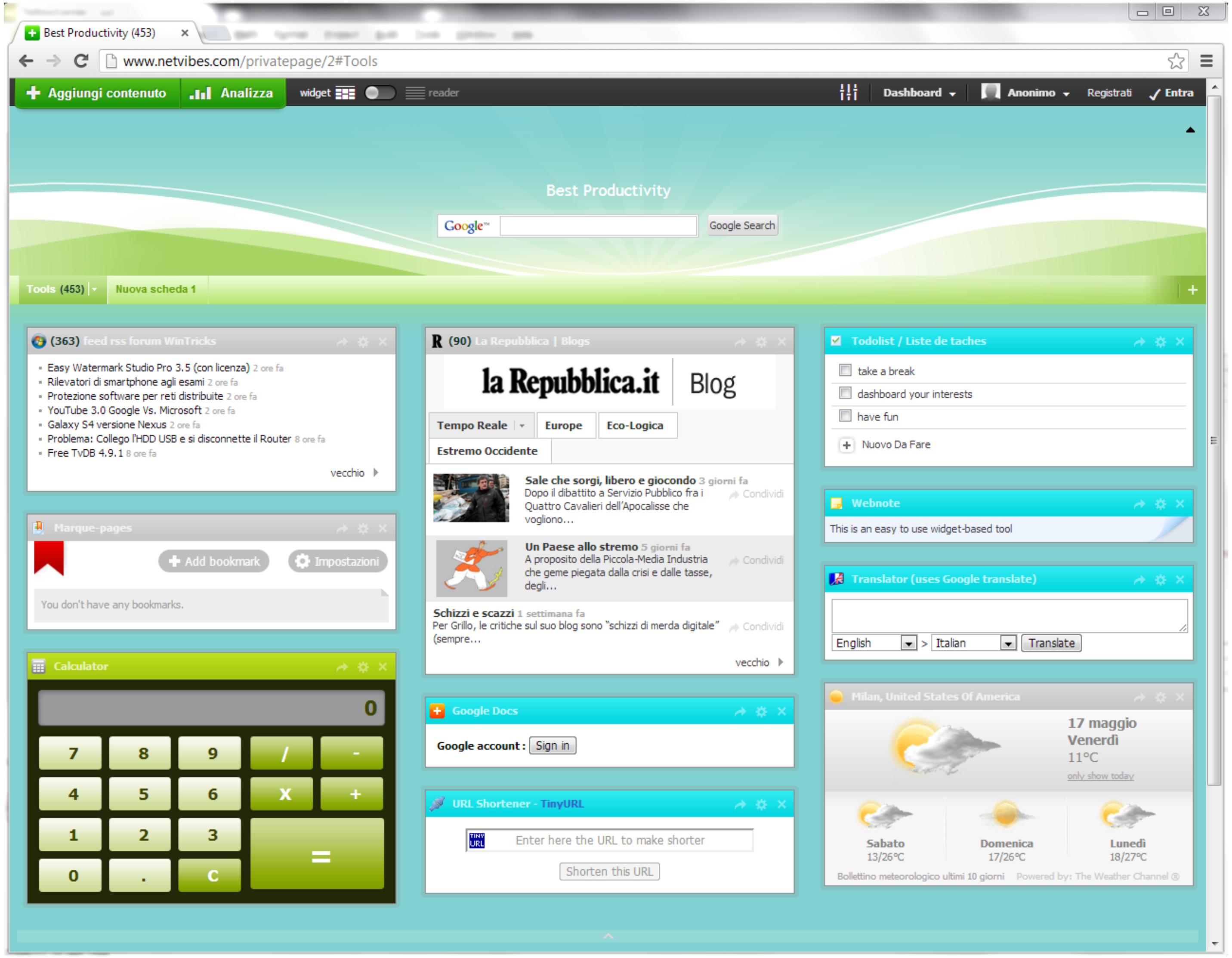

Fig. 6.14 An example of widget-based mashup created using the Netvibes platform. Different areas of the page correspond to different viewports, each one displaying the content of a different widget.

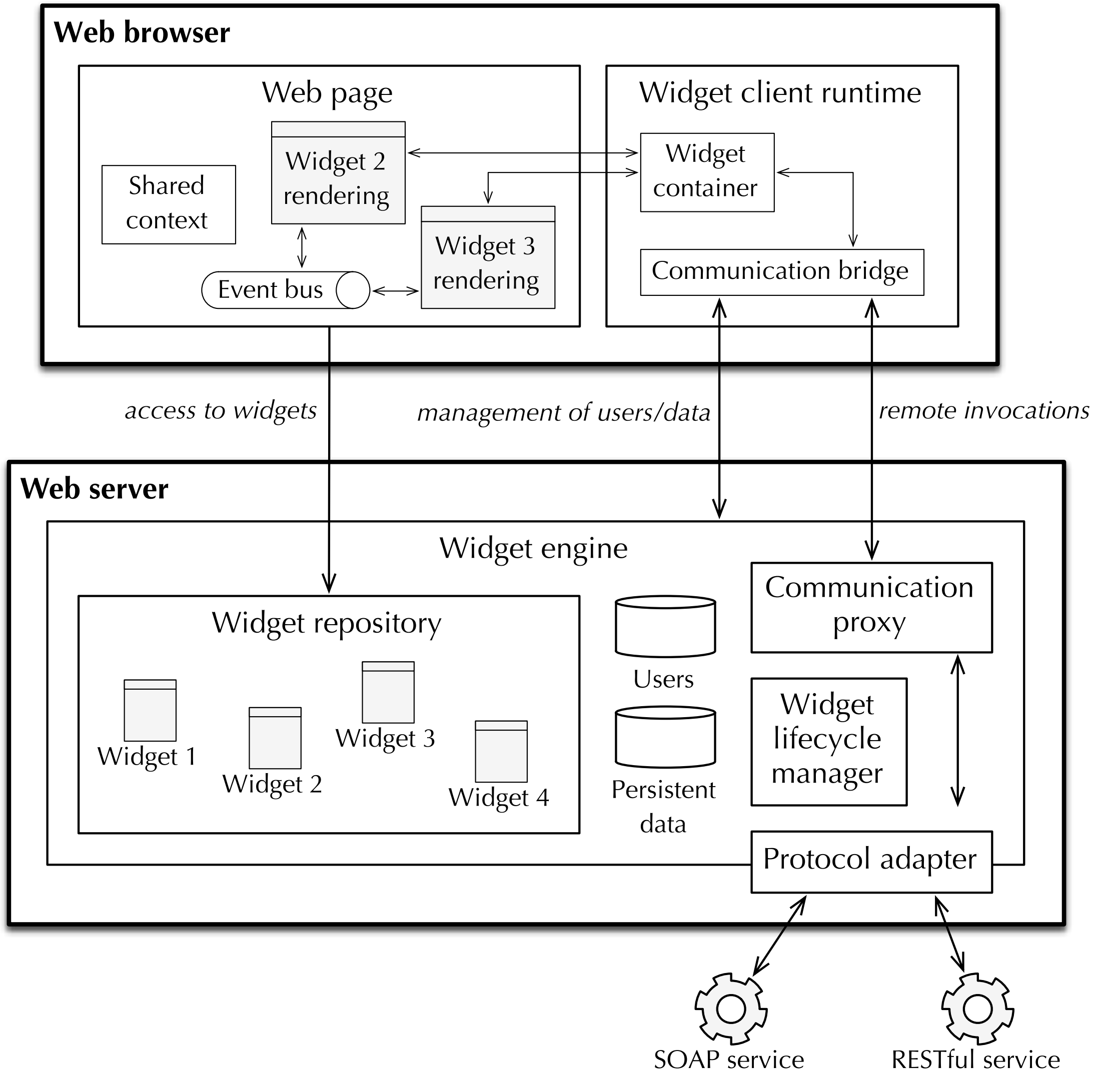

Fig. 6.15 Conceptual architecture of a widget portal with client-side inter-widget communication.

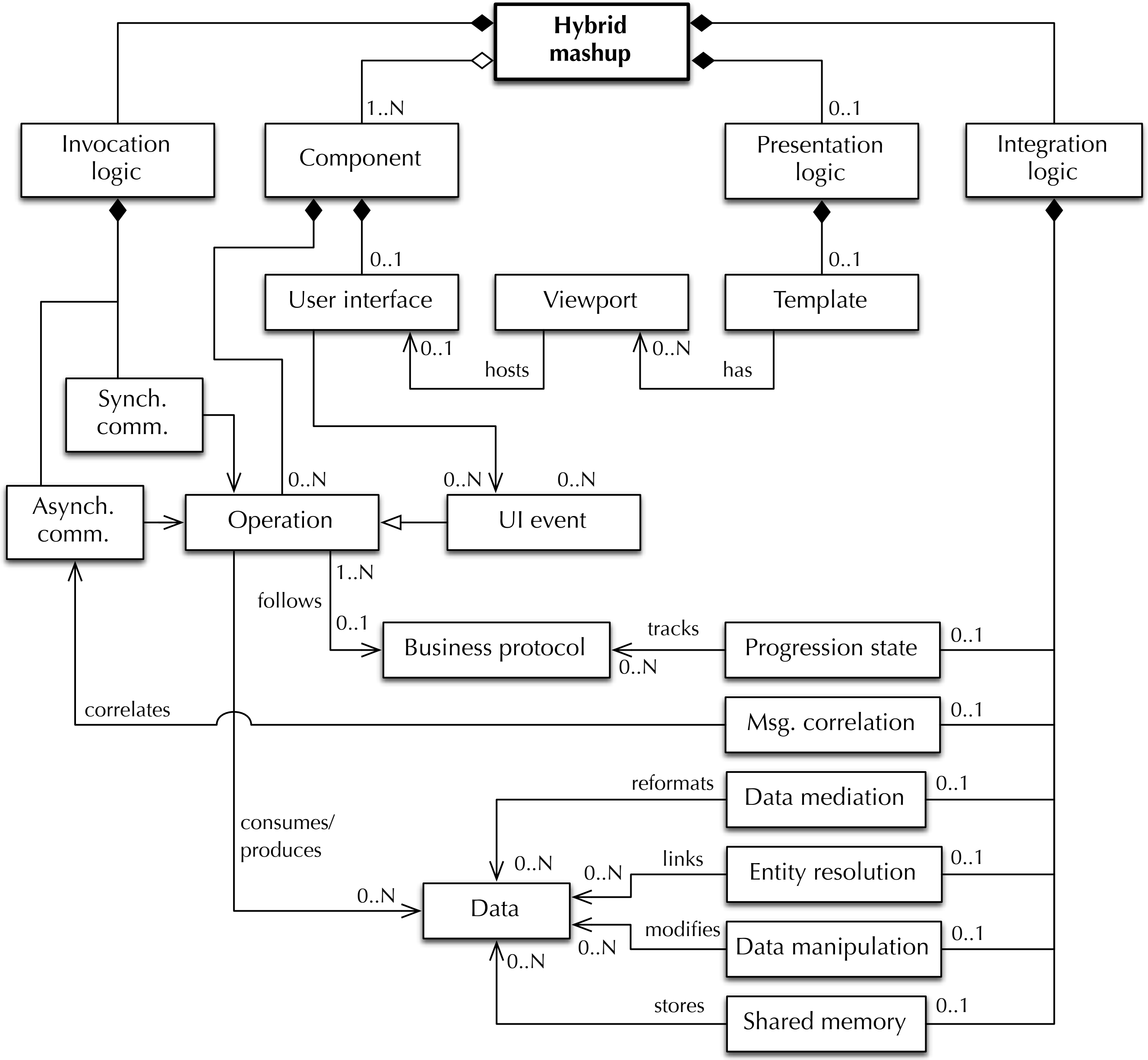

Fig. 6.16 The hybrid mashup model conciliating integration at the data, logic, and presentation layer.

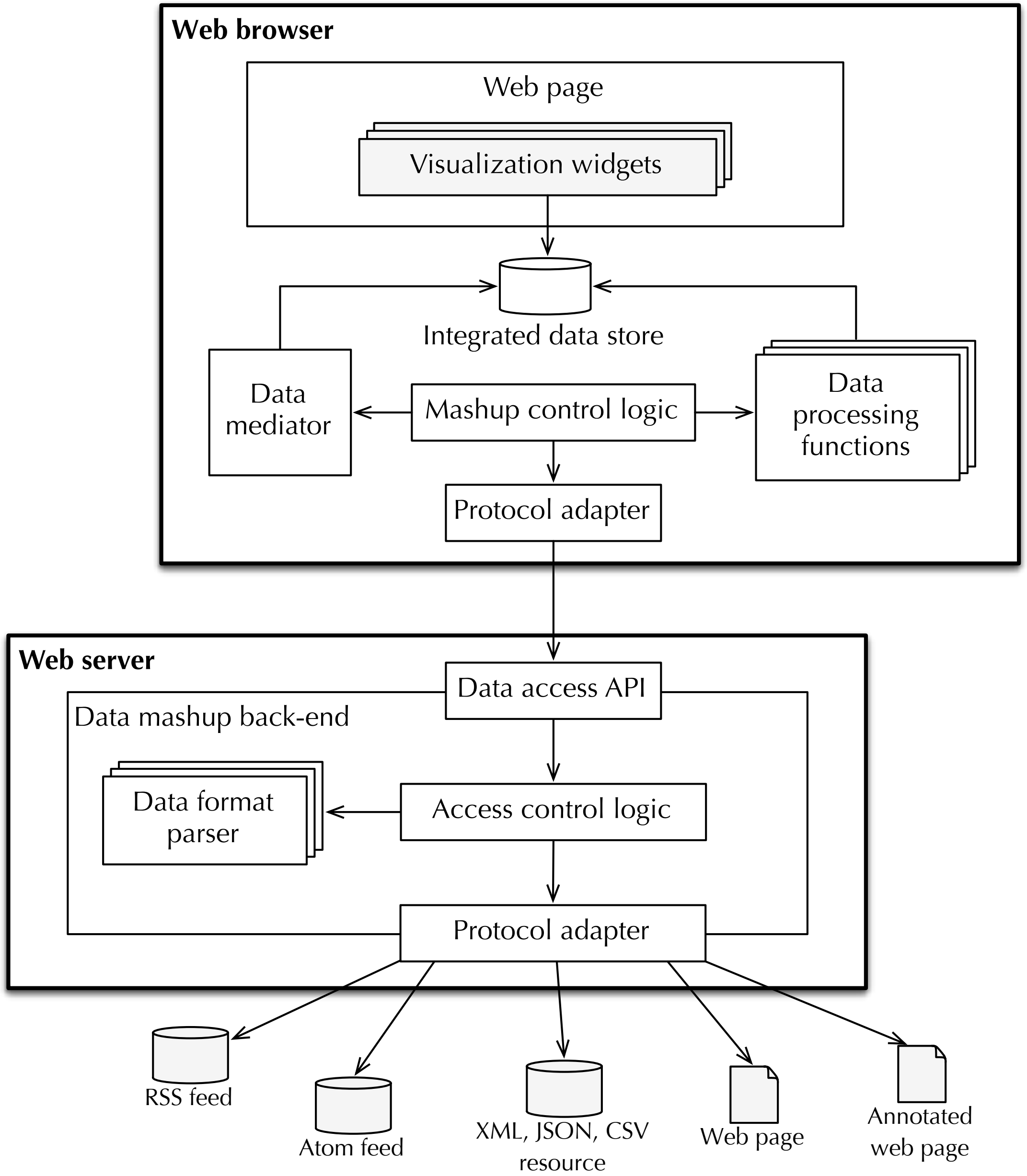

Fig. 6.17 Architecture of a data mashup with client-side integration logic and visualization (we omit the client-internal details regarding UI management).

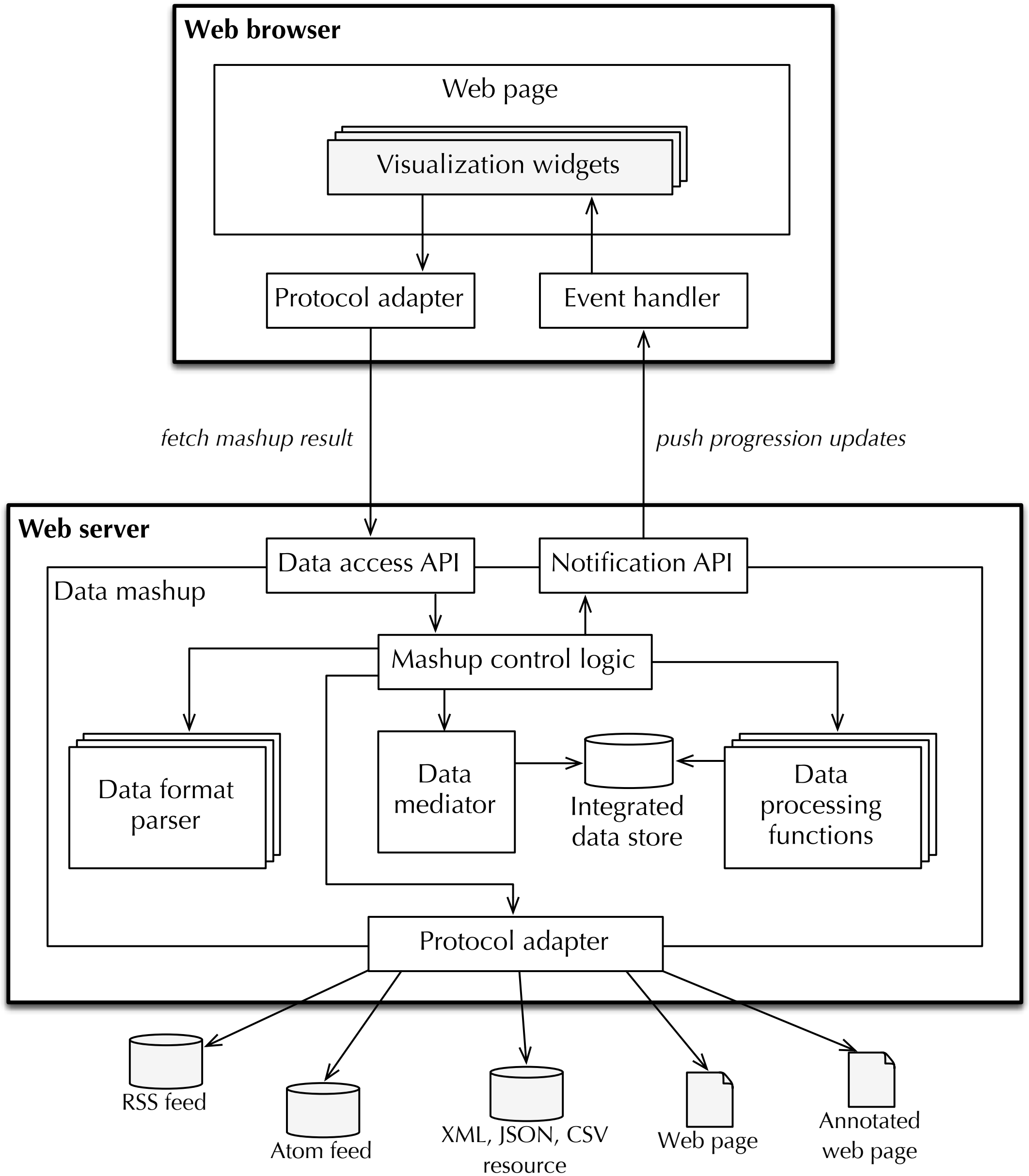

Fig. 6.18 Architecture of a data mashup with client-side rendering of results and possible intermediate progression information.

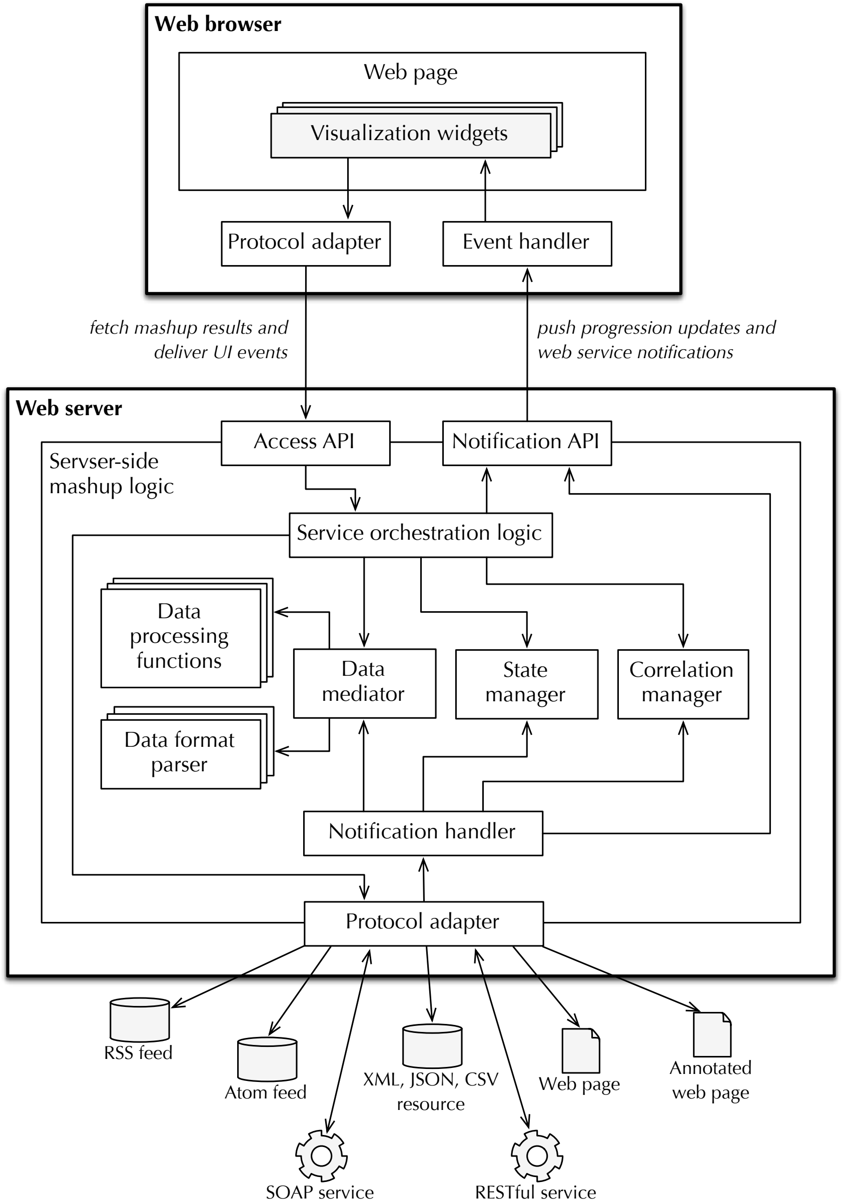

Fig. 6.19 A possible architecture for a universal mashup, integrating web services and processing data on the server side while using custom UI widgets for the visualization of results and the interactive control of the mashup.

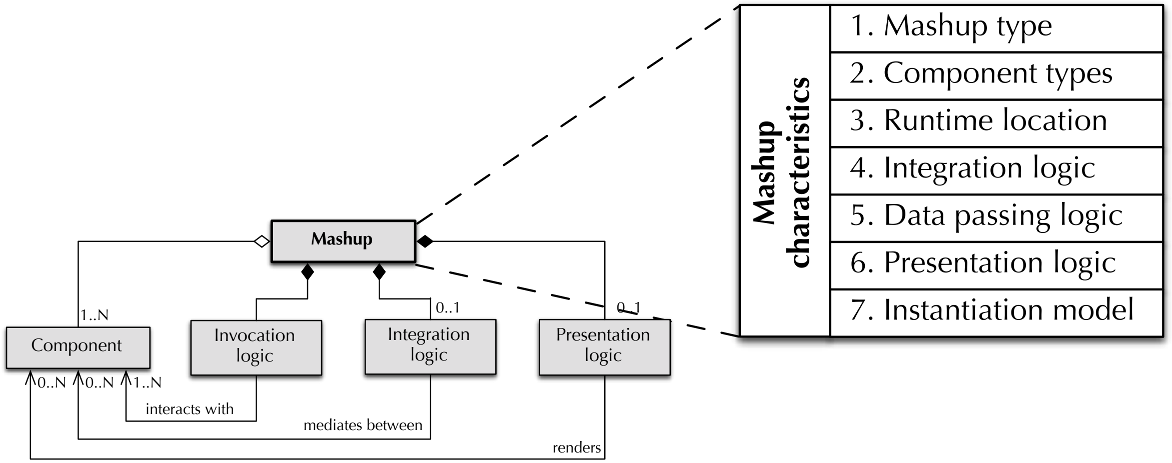

Fig. 6.20 Seven characteristics to distinguish different mashup models.