

## Abstract

iMeteo is a web-based weather visualization tool. Designed with an extensible J2EE architecture, it is capable of displaying information from heterogeneous data sources such as gridded data from numerical models (in NetCDF format) or databases of local predictions. All this information is presented in a user-friendly way, being able to choose the specific tool to display data (maps, graphs, information tables) and customize it to desired locations.

### Modular Display System

Visualization of the data is achieved through a set of mini tools called widgets. A user can add them at will and arrange them around the screen easily with a drag and drop movement. They can be of various types and each can be configured separately, forming a really powerful and configurable system. The *Map* is the most complex widget, since it can show several variables simultaneously (either gridded or point-based) through a layered display. Other useful widgets are the *Histogram* plot, which generates a graph with the frequency dist of distribution variable and the *Timeline* which shows the time evolution of a variable at a given location in an interactive way.

### Customization and security

Following the trends in web development, the user can easily customize how data are displayed. Due to client side programming based on technologies like AJAX, the user's interaction with the application likewise application desktop based on rapid response times. When a user is registered, then he can also save his settings in the database, allowing access from any other Internet terminal using his profile. There is particular emphasis on application security. The administrator can define a set of user profiles, for authorization access to certain data sources, geographic areas or time intervals.

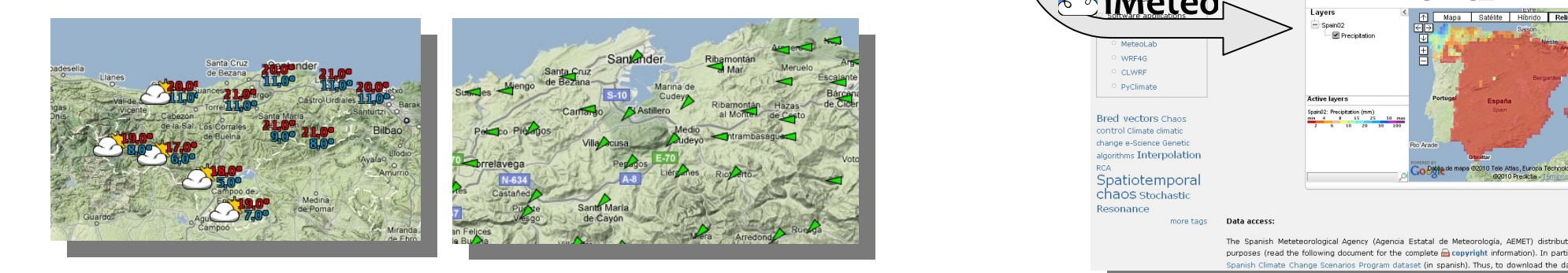
## iMeteo key features

### Software architecture

- Portable J2EE application
- Modular architecture following MVC design pattern
- Based on Open Source frameworks such as Spring, Struts or iBatis
- Authorization and authentication with Spring Security. It's based on user roles and allows to restrict content access (data sources, geographical areas, meteorological variables, widgets) and actions
- Fully internationalization (i18n) via Struts2
- Non-intrusive caching is provided by EHCACHE configurable in XML files.
- Fully customizable visualization

### Visualization

- Widget-based visualization: maps, tables and graphs (temporal series charts, histograms, meteograms, etc.)
- API to embed widgets on external webs via an *iframe* HTML tag
- Custom visualization for each meteorological variable specifying different Java classes



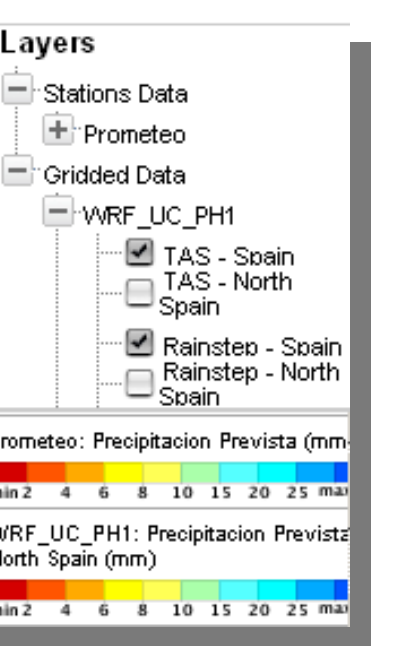
### Data sources and formats

- Data sources
  - Relational databases via JDBC access
  - Local or remote file based systems (for example, via HTTP servers and from OPeNDAP servers)
- Data formats
  - Gridded data: iMeteo relies on the Unidata Java NetCDF interface to read netCDF and GRIB files.
  - Observations data stored in relational databases (customizable formats)
  - Other file-based custom format (for example, Meteolab observations format: <http://www.meteo.unican.es/software/meteolab>)

## Visualization: the map widget

The map widget is the main visualization tool. Meteorological data is represented on the map as a layer. A layer might represent a NWP model output variable, such as temperature or rainfall, a particular type of data such as a satellite image, or point observations data. A layer defines how to display the data it references and where is located.

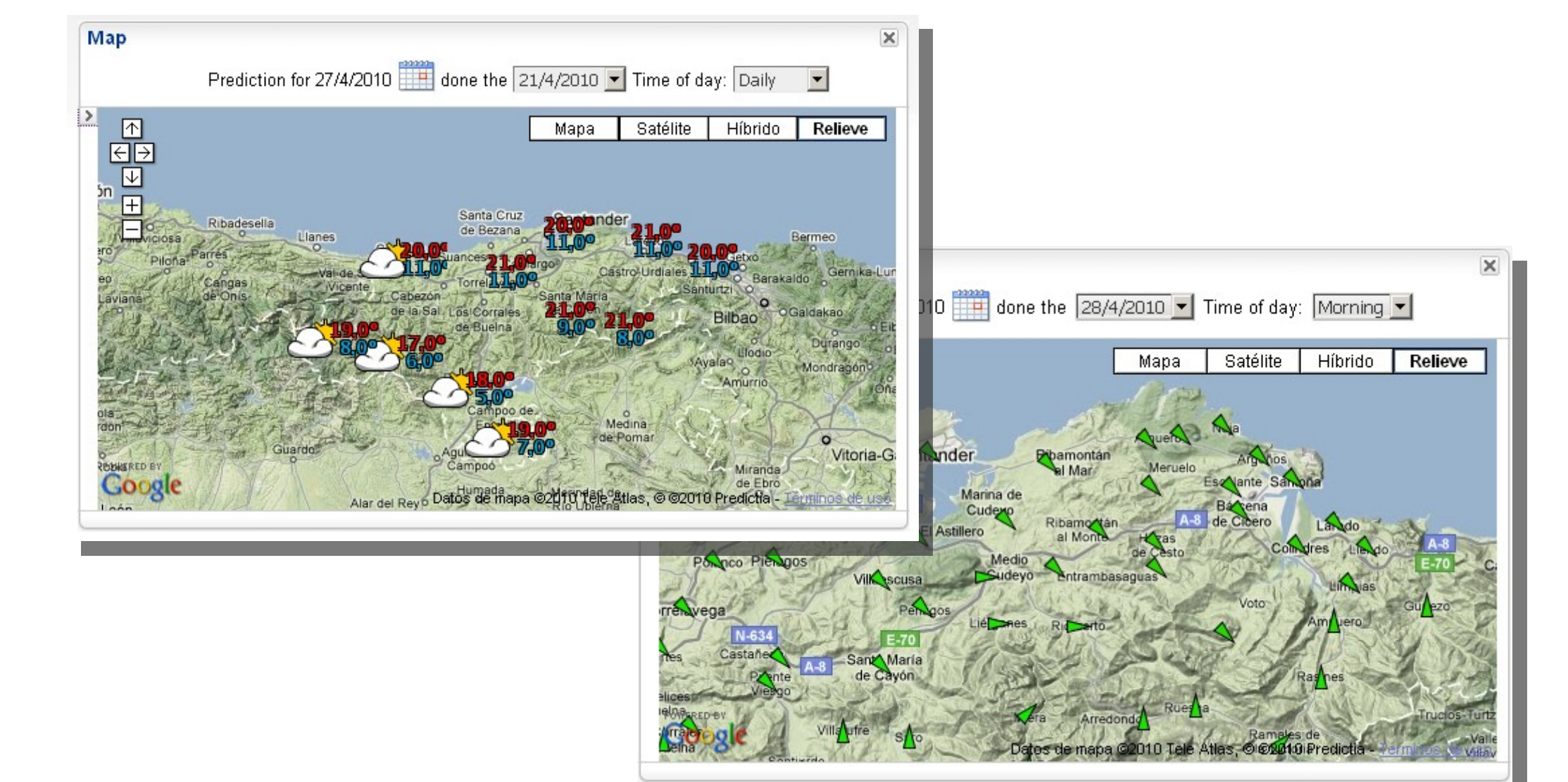
A map can display several layers simultaneously controlling the position and opacity for each one with the layer tree



Stations may have an associated weight, making it possible to be represented in a clustered-form, showing only the most important ones at distant zoom levels, and increasing the number as we approach.

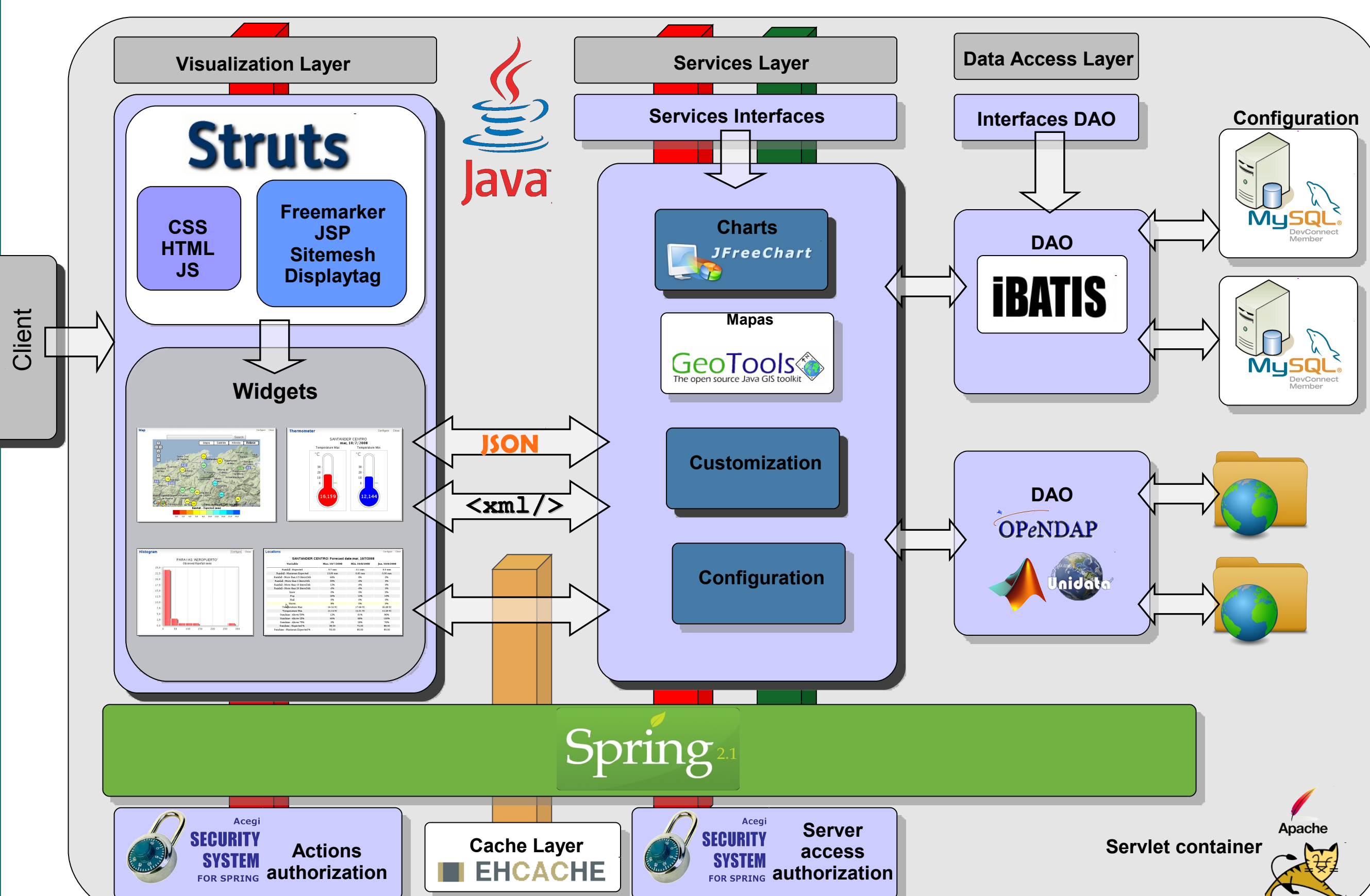
Sometimes, to display a layer it is necessary to read more than one variable from the data source. For example, wind direction is usually stored in two separate variables (U and V components). To manage this problem, iMeteo allows to define virtual variables calculated from the original ones.

The next figures show examples with a layer displaying together three variables: cloudiness, maximum and minimum temperature (up) and a wind surface layer (down):



## Architecture

iMeteo is a web application written in Java. It is deployed as a standard Java WAR file into a Java Servlet Container such as Tomcat. iMeteo follows a MVC (model-view-controller) architecture to separate the business logic and data of the application from the presentation of data to the user. The controller layer is provided by the framework Struts2, the view is based on FreeMarker and JavaServer Pages while the model is implemented as Java classes. Spring Framework is used to support Inversion of control (IOC) resolving component dependencies and authorization via Spring Security Framework. Finally, iBatis provides the data mapping process (Object-Relational mapping) and a persistence API while EHCACHE is used to cache frequently accessed data in a non-intrusive approach.

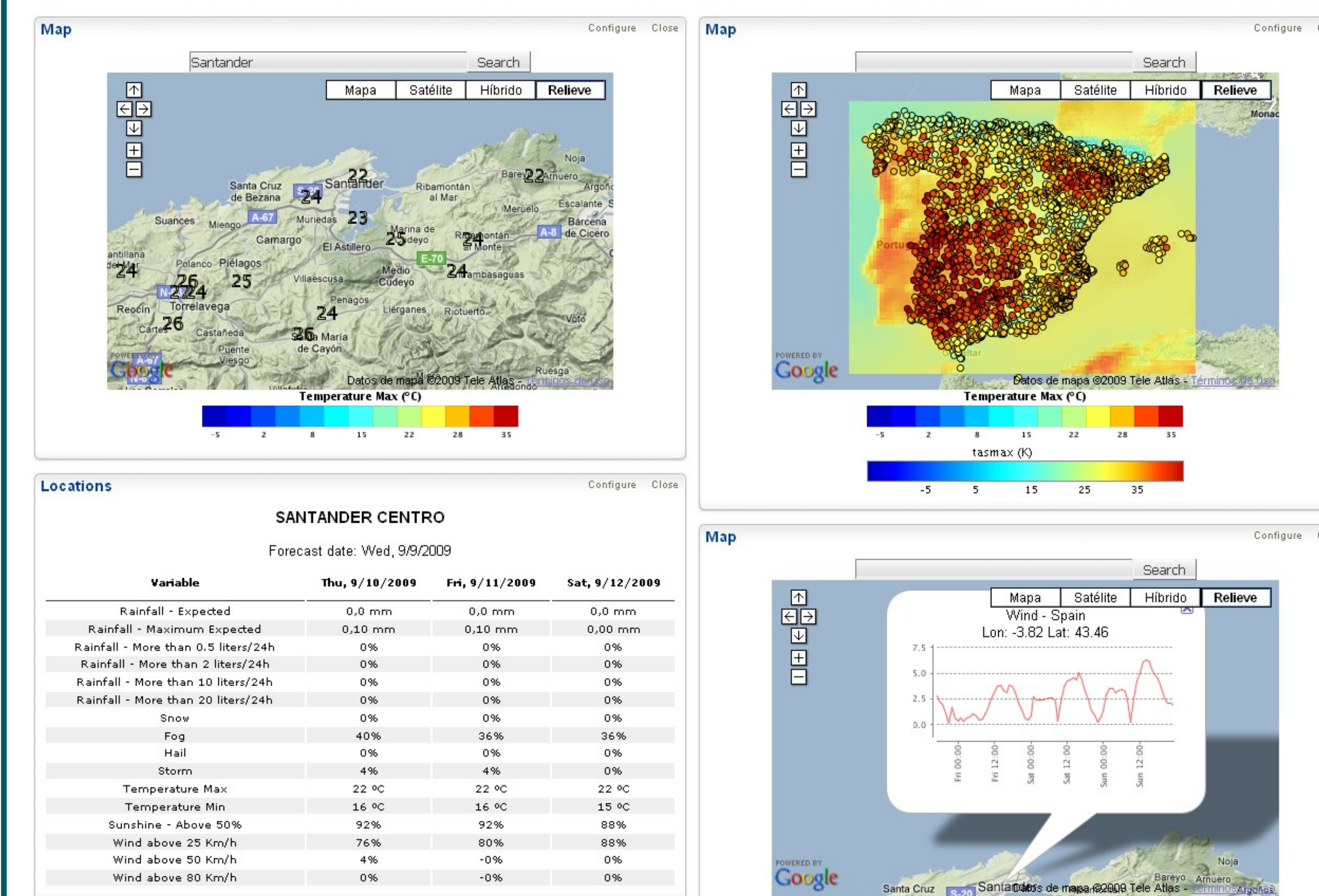


iMeteo can be easily extended to incorporate new data sources (relational databases or file system based) or different widgets which are views on content that can be customized. By default, iMeteo provides access to GRIB and NetCDF files served by an OPeNDAP server (based on UNIDATA's NetCDF-java library) and relational-databases. Widgets based on maps, tables and graphs are used to visualize data.

## Front-end

iMeteo front-end is a customizable AJAX-based dashboard or personal meteorological web portal. It is based on configurable widgets that can be added to the dashboard via Drag&Drop. It is built on YUI (Yahoo User Interface) for easy customization and browser compatibility. Every widget instance can be configured, minimized and maximized individually. User personal configuration is stored on a MySQL database.

iMeteo follows a *one-widget-multiple-instances* design allow the same widget to be displayed several times on the dashboard (most likely with different settings).



iMeteo provides an API (Application Programming Interface) to include widgets on external webs via an *iframe* HTML tag. Using this API, an URL is provided to embed easily a customized widget instance in any web.

## References

- [1] Donald Brown, Chad Michael Davis, and Scott Stanlick. "Struts 2 in Action" Manning, 2008.
- [2] "Struts 2 Documentation" 2009 <http://struts.apache.org/2.x/docs/home.html>
- [3] "The Spring Framework - Reference Documentation" 2009 <http://static.springsource.org/spring/docs/2.5.x/reference/index.html>
- [4] "Ibatis Documentation" 2009 <http://svn.apache.org/repos/asf/ibatis/java/ibatis-2/trunk/ibatis-2-docs/en/>
- [5] "JavaTM 2 Platform, Standard Edition, API Specification" 2009 <http://java.sun.com/javase/6/docs/>