MIDDLE EAST TECHNICAL UNIVERSITY

COMPUTER ENGINEERING DEPARTMENT

Senior Project
Initial Design Report

A Company Manufactures Everything
(ACME)

TABLE OF CONTENTS

1. Introduction

The Internet has become an indispensable part of our life in conjunction with the growth of technology. People of all age groups and companies in all sectors use the Internet extensively. Therefore, current time period is also called as Information Era. Web is the most contributive element to this fame.

In recent years, web technology has made big progress. At 2004, O'Reilly and MediaLive International have organized a conference with contribution of big world-wide web companies such as (Google, Yahoo, Msn, Amazon, eBay…) to discuss future of the web and to give a name to recent developments and trends. At this point a new decision has aroused, Web 2.0. The complex and evolving technology infrastructure of Web 2.0 includes server-software, content-syndication, messaging-protocols, standards-based browsers with plug-ins and extensions, and various client-applications. These differing but complementary approaches provide Web 2.0 with information-storage, creation, and dissemination capabilities that go beyond what the public formerly expected of websites[1]. And AJAX is the one of the technologies coming with this concept. AJAX is not Web 2.0; it is a one of the techniques featured in a Web 2.0 website. Web technology has widely adopted the Web 2.0, every day number of new web 2.0 pages increases, therefore number of pages with AJAX increases. AJAX that stands for Asynchronous JavaScript and XML is a new technology that helps to create interactive web applications. In this technology there is no need to reload the whole page, instead only the necessary and sufficient parts of the web application will be reloaded[2]. So we can say AJAX technology brings the power of internet and the speed of desktop applications together[3]. Therefore, web pages with AJAX technology are mostly preferred by users who are got familiar with AJAX technology by the web pages generated by Google Company such as Gmail, Google Maps.

[1] Web 2.0, Wikipedia, the free encyclopedia http://en.wikipedia.org/wiki/Web_2
[2] *AJAX (programming)* http://en.wikipedia.org/wiki/AJAX_%28programming%29
[3] *Mastering AJAX, Part 1: Introduction to AJAX* by Brett McLaughlin April 12, 2006, http://www.javascriptsearch.com/guides/Beginner/articles/MasteringAJAX_1.html

Today implementing web pages with AJAX technology saves time of end-users by increasing efficiency of the web-pages by shortening loading time of the page. However, developers are losing time when they are trying to implement this kind of web applications. Nowadays, speed is a priority for developers, and most developers are using environments that help them to create their applications in a more rapid way. So most developers prefer integrated development environments to create their applications.

2. Project Definition

As AJAX is a new, powerful technology, developers need such a powerful environment which simplifies code writing and makes it easy to design web sites using AJAX.

LADES will be an AJAX Development Environment Studio, which will be able create any (AJAX based or not) web application. LADES will be a platform independent and a user friendly development environment. In its design, diversity of users, their different knowledge levels, different devices on which it runs, different ways of interacting, different implementations and environments will be all taken into consideration. In LADES, simple tools that allow rapid development and rapid deployment will be available. Also, it will simplify access to databases from user interfaces as easy as drag and drop. In order words, our project LADES will be a rapid web application development environment that will let you create AJAX web page applications while reducing complexity and deployment times.

2.1. Application Areas

Nowadays we know the importance and frequent use of web applications. As well as web applications AJAX usage frequency is getting higher so that the web applications and AJAX has a wide usage area. Consequently our LADES program has several application areas. Most common of these application areas are as the following:

- Mapping applications
- Real time search engines
- Chat programs
- Real time form validation and processing
- Online shopping applications
- Email applications
- And any kind of web applications

In conclusion, when easiness and efficiency of our LADES program is taken into consideration, AJAX can be used in web any application; both for small and complicated ones. Therefore, we can say our LADES program will be helpful for most of the web application developers. No matter the complexity of the web application, as developers prefer writing less code, and saving time; our project will provide these features to the web application developer.

2.2. Project Features

Some features that are listed through this list will compose the core functionality that is required for basic functional AJAX development software. Other features that will extend the usability of the product will be marked as optional and will be implemented as external components. These components will be added to project if the core functionality works properly and we have enough time left from implementing core functionality.

In addition to this, we would like to implement a modular design in our project. If we can achieve this, we can implement the core functionality and add new modules to the system any time we want. We will work on this modularity issue.

LADES begins like many other applications, by starting a new project by the help of wizards. A project defines the bunch of the web pages which serves one aim or just a single application. In addition, using project templates is another option to get started.

To see the details of project, a navigation tool provides an organized view of the objects in a view, as well as commands for managing a project. Alternatively, the File System view could be selected, which reflects the physical file system of the project on disk.

After that to create your application, there are two options,

- **Drag-and-Drop Visual Components:**

  A simple drag-and-drop user interface design ensures fast refinement from prototype to completed application. In addition to creating and using visual components like text fields and buttons, you can also assign database information, AJAX events to components through drag-and-drop. Component size and placement can be adjusted using the mouse or keyboard. Also by mouse or keyboard interaction views or properties of these visual components can be modified by the help of LADES only filling a text boxes. In addition, component size and placement can be adjusted by dropping components to desired location.

  When you open the application and start a new project, an empty canvas and a visual editor are presented. The canvas provides the work area for laying out the user interface. Applications can be visually built quickly and intuitively by dragging and dropping graphic elements and user interface components onto the canvas.

  At the top of design window, a toolbar is designed to provide access to graphical user interface components, such as custom graphics, HTML elements, database connection or adding AJAX events. Simply drag and drop graphic elements and user interface components from the toolbar onto the canvas constructs the web application by the default property of the components. Each component has properties that can be easily edited through their property fill wizard, which are available for standard components and for custom components. Typical properties might include text size and styles, color, or the size of a component.

  In addition, views can switch from visual to working with code directly. Synchronized editing ensures that changes made to objects in any view are reflected in the canvas.

- **Creating Manually:**

  For those of you who already know the programming language and prefer to hand code, or for those who want to see what's happening behind the scenes, LADES provides a Code Editor that supports HTML pages, PHP pages. In this mode, syntax highlighting and code completion is supplied to the user. Syntax colorization helps the user keep coding issues straight. Also the user can move between development modes, from visual to the Code Editor, with ease. Maybe we could add drag and drop functionality to this component, to supply code template for most used components.

LADES also provides different variety of functionality, such as debug, preview mode or running in browser or deployment of created files. LADES will be explained in detail in the following sections.

General features that LADES will support are as follows.

Design View will support:

- Drag & drop webpage components (core)
- Dragging webpage components to support repositioning and resizing (core)
- Setting webpage component properties and their events using its GUI (core)
- Generating AJAX related codes automatically (core)
- Creating database connections by user friendly wizards (core)

Text Editor will support:

- On the fly syntax highlighting (core)
- Syntax checking (optional)
- Auto indenting (optional)
- Find and replace support (core)
- Class / Function hierarchy browser (optional)

General project features:

- CVS support (optional)
- FTP upload support for remote deployment (core)
- Running the project on the remote machine (core)
- Timer based auto save (core)

2.3. Newly Added Features

The following features has been discussed among the project members and added to the core of the system after the analysis phase of the project. These can be explained as following:

- **Flexible Layout Design:** Each sub-window of LADES now is draggable. These subwindows are called 'Views' and these views can be docked to other panels, undocked as separate windows, put in split panels or some other tabbed panels. Also, different layout designs can be saved and loaded in run-time. By the help of this layout system, user has the full control over the views of his/her workspace. This layout system is explained in more detail at 6.1 Layout Design.
- **Multi-lingual Support:** LADES will now provide multi-language support. At the first stage, it will include English, German and Turkish language files. But it can easily be expanded to support new languages by adding new language files. Language of the whole system can be changed on the fly which is explained at 7 Multilingual Support.

2.4. Process Model

In our project we will develop our work by analyzing, designing, making implementation, and testing. So it is obvious that we will develop our project as a step by step manner. Then as a team we have come together and have thought about the possible process models that we may use in our project. We know that the project phases are distinct. The requirements are well defined and understood. But there is a possibility that in some phases we may do mistakes and sometimes we may need to return to a specific

phase and make changes. So during the project a feedback mechanism is needed. By the help of this information as a team we decided to apply “waterfall model with feedback” in our project.

“Waterfall model with feedback” suits our goal best, because the steps are distinct as we needed. In addition to this between each phase feedback loops exist so it allows us to make modifications easily.

2.5. System Requirements

2.5.1. Hardware Requirements

Our hardware requirement is composed of three main groups; first one is development side requirements, second one is user side and the last one is server side requirements.

Developer Side

For fast and comfortable development, approximate minimal requirements are as follows:

- P4 2.0 GHz or equivalent processor
- 512 MB RAM
- 40 GB HDD
- 32 MB Video Card
- Internet Connection (for testing purposes)

Computers of the project members meet the minimal requirements, so we don’t expect to face any difficulty in this area.

User Side

The user of our project “Lades” should a have computer capable of running several service programs if he/she wants to deploy his/her project locally (to his/her computer). Internet connection is required if the project will be deployed to a web server using ftp connection. Approximate minimal requirements, based on an assumption that the user will use local deployment, are as follows:

- P4 1.6 GHz or equivalent processor

- 512 MB RAM
- 40 GB HDD
- Internet Connection (for external deployment)

Server Side

Our server component is a typical web server. It will run an Apache web service with PHP support. PHP will have embedded MySQL libraries. According to users' choice the server running Apache may run MySQL server component too. This will increase the strain on the server thus will increase minimal requirements. The minimal requirements for full web service support are as follows:

- P4 3.2 GHz or equivalent processor
- 1024 MB RAM
- 100 GB HDD
- Internet Connection (broadband connection preferred)

2.5.2. Software Requirements

Our software requirement is composed of three main groups; first one is development side requirements, second one is user side and the last one is server side requirements.

Developer Side

During the analysis, design, implementation and testing phases of the project, we will use several tools. These tools can be divided into two main groups; first one is documentation tools and the second is development tools.

Documentation Tools

During the first term, we will be using several documentation tools extensively. Our choices for word processing are Microsoft Office Word combined with Adobe Acrobat Professional (for converting to PDF), or OpenOffice 2.0 Writer. We will usually use MS Word.

SmartDraw will be our primary drawing tool for diagrams because of its ease of use. In requirement analysis report we have drawn data flow, state transition, use case diagrams with SmartDraw. Gantt chart will be drawn with Microsoft Project.

Development Tools

For our project to be platform independent we have chosen Java as our programming language. In order to program in Java we need Java Development Kit (JDK). JDK is composed of two main parts; Java Runtime Environment (JRE) and command line development tools (such as compilers, debuggers, applets... etc). We will mainly use some Integrated Development Environments (IDE) to develop Lades in Java. These IDEs will be NetBeans and Eclipse. These IDEs are mostly interchangeable.

To test our project we need the web services that will be used in the web project that was created with Lades. These are Apache with PHP support and MySQL server.

User Side

For the user to run our project he/she will need JRE preinstalled in his/her computer Apache, PHP and MySQL must be present in his/her system if he/she chooses to deploy his/her project locally.

Server Side

Required software for the server side to operate correctly will be Apache, PHP and MySQL.

3. Modeling

3.1. Functional Modeling

3.1.1. Data Flow Diagram

3.1.1.1. DFD Level0

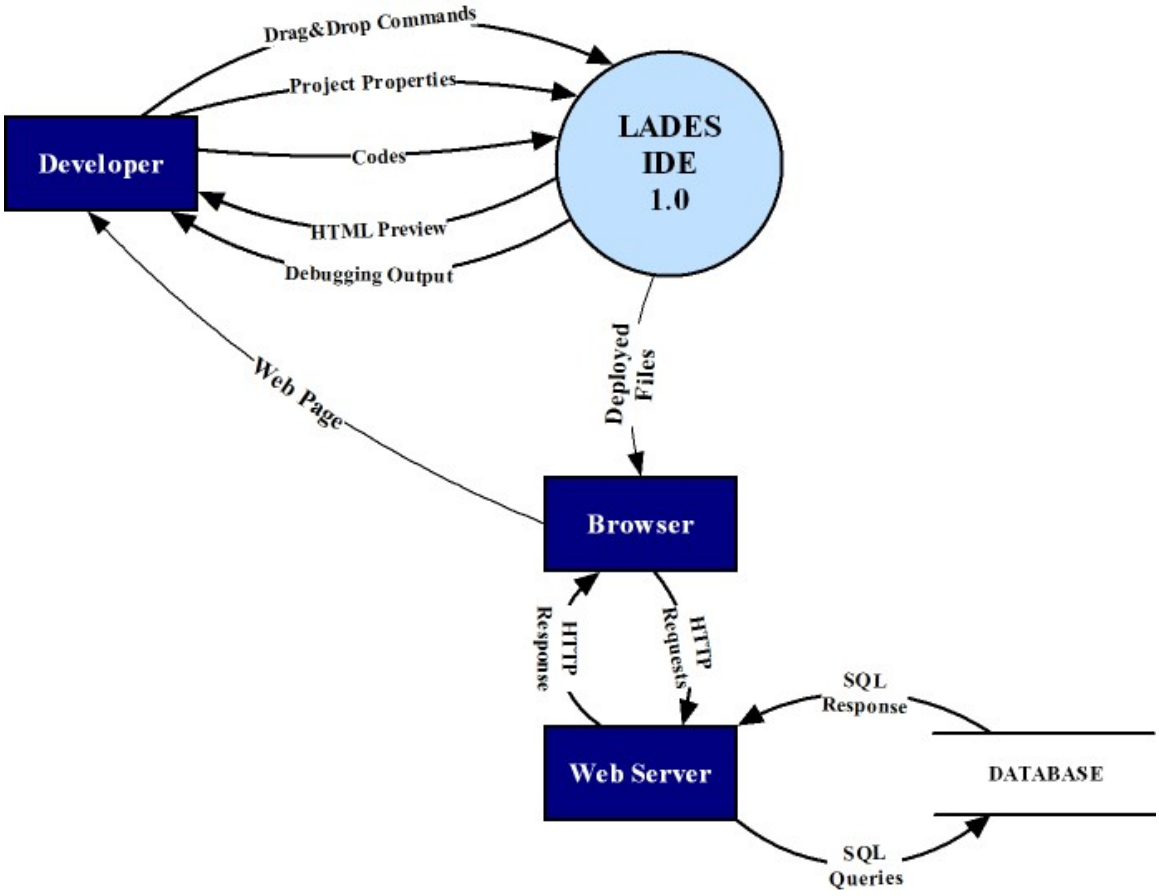


Figure 1 - DFD Level0

3.1.1.2. DFD Level1

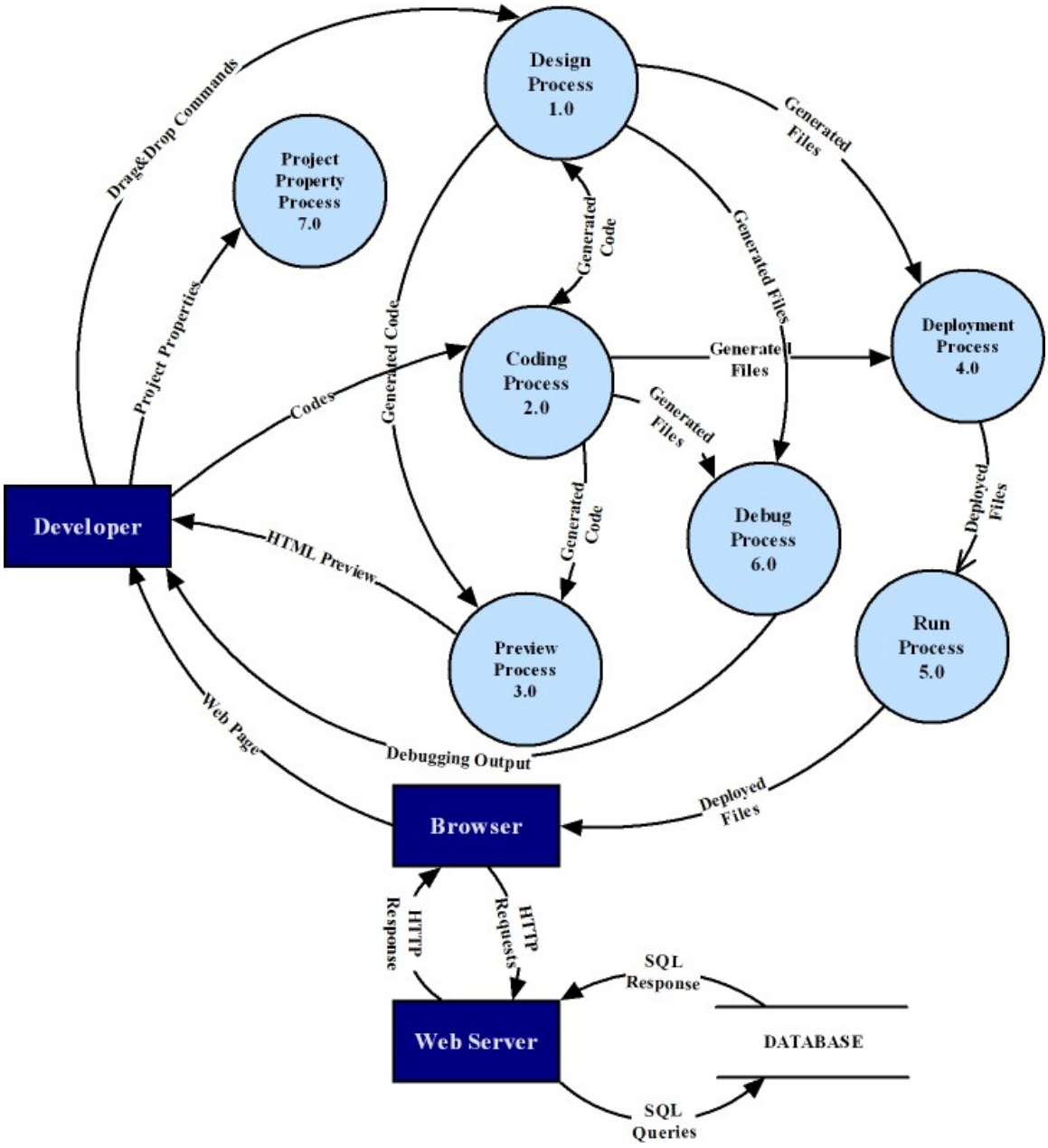


Figure 2 - DFD Level1

3.1.1.3. DFD Level2

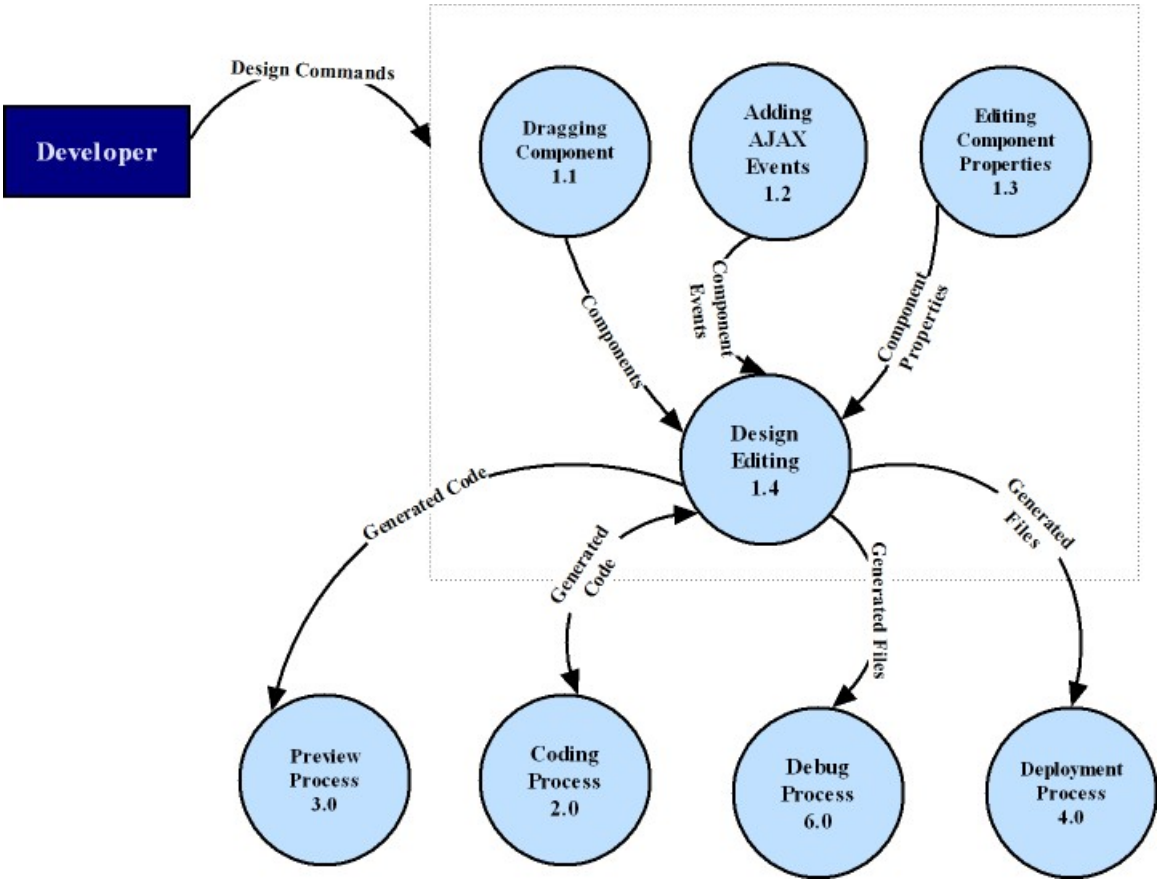


Figure 3 - DFD Level2 of Design Process 1.0

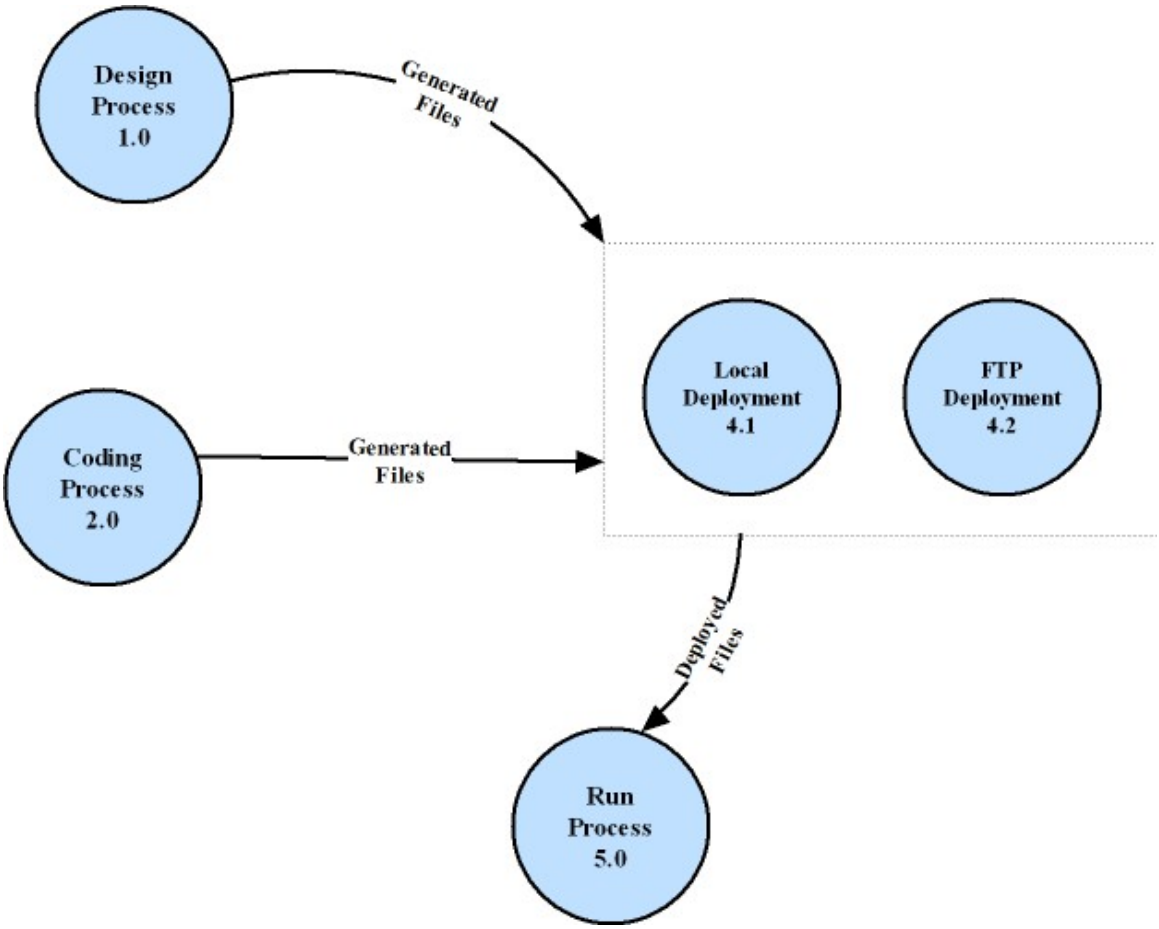


Figure 4 - DFD Level2 of Deployment Process 4.0

3.1.2. Data Dictionary

name:	Codes
where used / how used:	Codes are *input* to Coding Process and typed by User This data are validated and processed in coding Process such as syntax hightlighting and shows available options such as when a tag as opened a closing tag will be added and cursor will be moved to between tags.
Description:	codes = string

name:	Component

Where used / how used:	Component is *input* to Design Editting Process (1.4) coming from Dragging Component(1.1). General name of web page components which are used to create form with buttons, checkboxes or to add images or to connect to database or to add table.
Description:	Codes = string

name:	Component Event
Where used / how used:	Component Event is *input* to Design Editting Process(1.4) coming fromAdding AJAX Events(1.2). This Events are used to provide asynchronization which are used in onChange or onSelect JavaScript events.
Description:	Code

name:	Component Properties
where used / how used:	Component Properties are formed in Editting Component Properties Process(1.3) and given to the Design Editting Process(1.4). This information varies according to the component for example if component is used for database connection the info will includes properties such as an endpoint address of the database connection, username and password .If it is HTML form element, it could include aligment, size, color etc.
Description:	Form Element Property = size+alignment+color Database = database address + username + password +schema name

Name:	Deployed Files
where used / how used:	Input to Run Process and Browser. Output from Run Process and Deployment . It is a ready to run file which is either on localhost or remote web server
Description:	HTML or PHP File

Name:	Debug Output
where used / how used:	Output from Debug Process shown to the user. In case of the error, debug output shows syntax error information.
Description:	Error message | Well Done

name:	Drag & Drop Commands
where used / how used:	Design Process(1.0) *input* given by User It includes drag and drop functionality output by the help of these commands design window and generated code can be formed.
Description:	DragCommand = Name of web page component + Name of the part where it is dragged Drop Command= Name of the web page component + Name of the part where it is dropped

Name:	Generated Codes
where used / how used:	Input to Design Process and Coding Process and Preview Process. Output from Desing Process and Coding Process It is used to show any output the user, such as preview, code or design of web page
Description:	This very generalized data name.It consists of variety of elements.

Name:	Generated Files
where used / how used:	Input to Debug Process and Deployment Process. Output from Desing Process and Coding Process It is different than the generated code by the way that the it is not part of any code, the completed file are deployed or debugged
description:	HTML or PHP File

Name:	HTTP Request
where used / how used:	Input to Web server Output from Browser. It is automatically generated by browser when the user want to run the web page.
description:	HTTPRequest

Name:	HTTP Response
where used / how used:	Input to Browser. Output from Web server It is response to requester via webserver.
description:	The HTTP response typically consist of an HTML document, but can also be a row-text file, an image or some other type of document; if something bad is found in client request or while trying to serve the request, a web server has to send an error response which may include some custom HTML or text messages to better explain the problem to end users

Name:	Project Properties
where used / how used:	Project Property Process (7.0) *input* given by User This data is used as to adjust the IDE prefererences and view to customize the user requirements and habits such as auto-indent and syntax highlight, show class hiearchy.
description:	This very generalized data name.It consists of variety of elements.

Name:	SQL Queries
where used / how used:	Input to database(7.0) Output from web server
description:	SQL query

Name:	SQL Response
where used / how used:	Input toweb server Output from database(7.0)
description:	SQL query

Name:	Web page
where used / how used:	Result of the project shown in browser
description:	HTML or PHP file

3.1.3. Process Specification

Design Process 1.0

Users of LADES have two options to generate HTML code:

- By using the design window,
- By using the text editor.

Design process provides several options the user in order to generate HTML code. These are as follows:

- Design window has the capability of displaying current active components on the webpage. User will be able to drag these components through the design window in order to change their positions.
- Webpage components like form elements (checkboxes, text fields, buttons…), database connection elements (database connections, SQL queries), charts etc can be added from the "Webpage Components Tree" by using drag and drop.
- Several component events can be selected from the "Events Editor" section. These events include JavaScript events and AJAX related events.
- Component properties can be edited through the "Properties Editor". This editor also includes built-in CSS editor.

Although design process generates the required code, user will always be able to this code through the text editor part of LADES which is explained in next section.

Coding Process 2.0

Users have another option to generate the code, by using the text editor LADES provides. This process and design process works in collaboration with each other. Users may write some code in text editor and switch to design process. Design process processes these codes and displayed the webpage components in its window. Then users may use the functionality design process provides. Vice a versa users can switch from design process to code process.

Preview Process 3.0

The generate code from design process and code process can be viewed by preview process. This process uses an embedded browser (Mozilla Firefox 2.0) to display the created webpage. Users may preview their web pages without deploying the files and opening and external browser.

Deployment Process 4.0

Generated files can be deployed on a local computer or on a remote web server. This process provides this functionality. Remote deployment uses the FTP protocol.

Run Process 5.0

Created webpage can be seen on an external browser at any time. This external browser can be selected through LADES preferences. By this process, user can see the output on any browser unlike the embedded browser support. It will have two different modes to display the webpage.

- Displaying on an external browser using the absolute file path
- Displaying on an external browser by using local deployment and running on the local web server.

Debug Process 6.0

LADES project supports debugging of JavaScript and PHP. Debugging support may detect syntax errors and watching variables.

Project Property Process 7.0

Several IDE options can be changed through this process. Hotkeys, default project settings and text editor settings like syntax highlighting, viewing line number, etc can be changed by the Preferences window.

Dragging Component Process 1.1

There is the list of draggable items. Dragging components process takes the design commands from the developer, forms the web components, after that this process sends related components to "Design Editing Process 1.4". By the help of this process developer can be able to add drag and drop web page components to web application easily.

Adding AJAX Event Process 1.2

As the web developer will be able to add AJAX events to strength the functionality and efficiency of program, the Adding AJAX Events process takes the design commands from the user, forms related events and sends the component events to the "Design Editing Process 1.4".

Editing Component Properties Process 1.3

Design commands are taken from the developer. Developer can edit the properties of the web page components, and when the editing component properties process takes the design commands from the user, send the component properties to the "Design Editing Process 1.4".

Design Editing Process 1.4

Design Editing Process takes drag and drop components from dragging component, component events from “Adding AJAX Events Process 1.2”, and component properties from “Editing Component Properties Process 1.4”. Design Editing Process displays the formed web components on the screen. This process also supports to change the location of the web form components. Codes and files are generated according to the gathered components from the related processes and the design editing process sends generated codes to “Preview Process 3.0”, sends generated codes to “Coding Process 2.0”, sends generated files to “Debug Process 6.0”, and again sends generated files to “Deployment Process 4.0”.

Local Deployment Process 4.1

If the developer prefers to run the process on local computer, “Local Deployment Process 4.1” takes place. This process takes the generated files from the “Design Process 1.0” as well as from the “Coding Process 2.0”, and then deploys the files to local web server root.

FTP Deployment Process 4.2

If the user wants to deploy the generated files to a server, “FTP Deployment Process 4.2” starts to perform. This process gets the generated files from the “Design Process 1.0” and from the “Coding Process 2.0”, and then deploys these files to remote web server root by using the FTP protocol.

3.2. Behavioral Modeling

3.2.1. State Transition Diagram

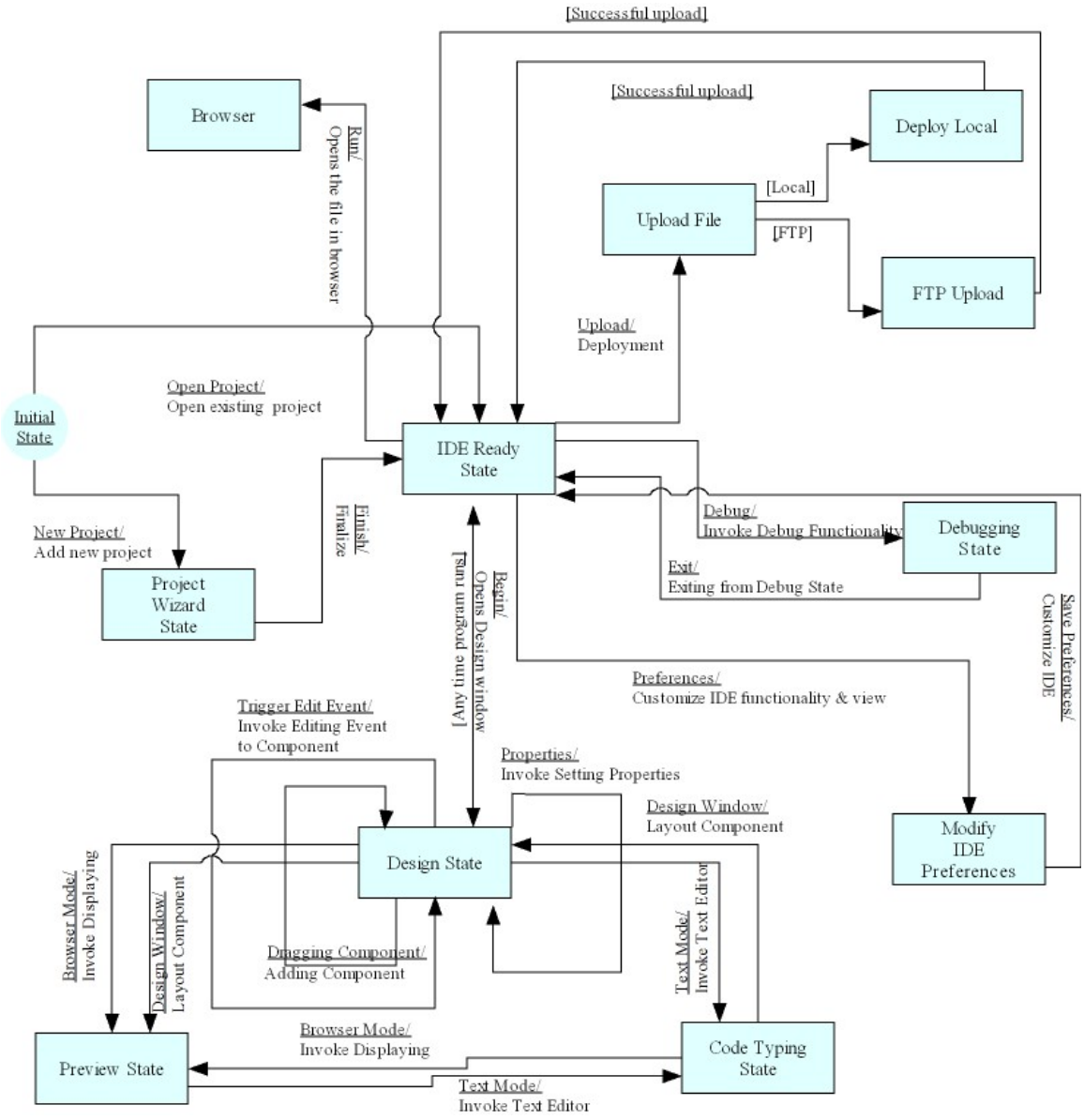
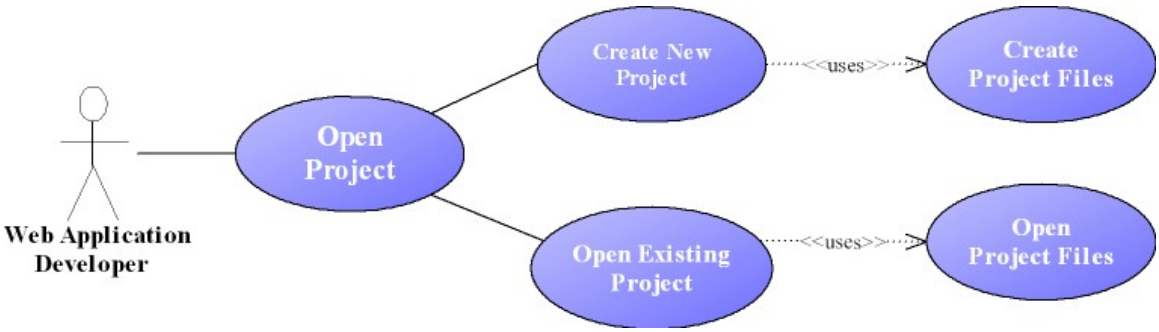


Figure 5 - State Transition Diagram

4. UML Diagrams

4.1. Use Case Diagrams

4.1.1. Open Project



Use Case: Open Project

Primary Actor: Web application developer.

Goal in context: To develop a web application with LADES IDE, the first step is to open a project.

Preconditions: To open a project in LADES IDE the program should be installed completely and correctly, and the specified requirements should be satisfied.

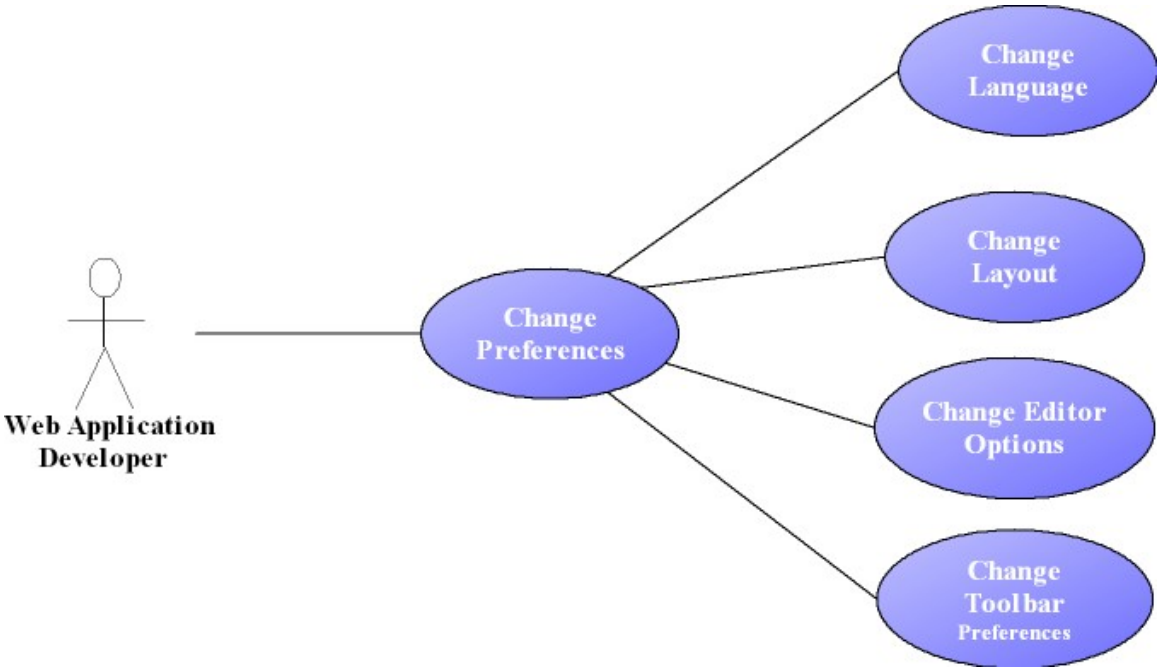
Scenario:

1. By using icons or menus the user is able to open a project.
2. Related to the type of choice (the user opens an existing process or create new project), the system displays related file dialog box.
3. The web application developer makes necessary arrangements.
4. The web application developer clicks ok button at the file dialog box.
5. Then project is formed.

Exception: Unrelated type of files with the LADES IDE causes exception.

In open project the user have two choices. One is to create a new project; the second one is to open an existing project.

4.1.2. Changing Preferences



Use Case: Change Preferences

Primary Actor: Web Application Developer

Goal in Context: To customize LADES application including text editor options like background and text-highlight color, layout arrangement, language of the menus, toolbar layout, etc.

Preconditions: Necessary packages must be installed a priori.

Trigger: In the first execution following the installation, system automatically starts "Change Preferences" process.

Scenario:

1. When Web Application Developer executes "Change Preferences" process by clicking buttons or pressing keyboard shortcuts.

2. “Change Preferences” process displays all possible functionalities of selected feature.
3. Web Application Developer selects or changes preferences displayed in the menu.
4. Web Application Developer confirms its selection by clicking confirm button.
5. System changes its configurations and change display mode.

Exceptions:

1. Changes are not possible values for the system – system displays appropriate error message and cancels the user request.
2. Web Application Developer selects “change to graphical design environment or textural design environment” – see use case: “Design with GUI” or “Design with Text Editor”.

Use Case: Change Language

Primary Actor: Web Application Developer
Goal in Context: Provide multiple language support for menu items.
Preconditions: Appropriate language packages must be installed a priori.

Scenario:

1. When Web Application Developer executes “Change Language” process by clicking button in the menu bar.
2. “Change Language” process displays all possible languages of the system.
3. Web Application Developer selects the language s\he wants from the list.
4. Web Application Developer confirms its selection by clicking confirm button.
5. System changes its configurations and change menu bar and toolbars to new language.

Exceptions:

1. Selected language file cannot be recognized by the system. System cancels the request and displays an appropriate error message.

Use Case: Change Layout Settings

Primary Actor: Web Application Developer
Goal in Context: Provide a customizable layout to user.

Scenario 1:

1. Web Application Developer can change position and orientation of all panels (file panel, component hierarchy panel, component libraries panel, properties and events panel, output panel, code panel, design panel and browser panel) by dragging with mouse.
2. "Change Layout Settings" displays possible new place of the panel
3. When Web Application Developer finishes dragging panel is translated to new location on the screen
4. System changes its configurations and displays new configuration of the panels.

Scenario 2:

1. Web Application Developer selects "save layout" or "load layout" from the menu bar.
2. If save mode is chosen, "Change Layout Settings" asks location to save the layout, or if load mode is chosen system ask for a layout from the user.
3. User selects layout to load or path for the layout to save and confirms selection.
4. System saves or loads the layout.

Exceptions:

1. There cannot be enough memory for the file to be saved.
2. For the layout file to be loaded there cannot be necessary rights for reading the file. At this situation layout cannot be loaded and an error message indicating the error is displayed.

3. Layout file to be loaded cannot be recognized as a layout for the system. Request is canceled and an error message is displayed.

Use Case: Change Editor Options

Primary Actor: Web Application Developer
Goal in Context: Font size, type, color; page borders, paragraph and tab indent, etc properties can be changed by the user.

Scenario:

1. When Web Application Developer executes “Change Editor Options” process by clicking button in the menu bar or pressing keyboard shortcut.
2. “Change Editor Options” make demanded functionality.
3. System changes its configurations and change editor according to user request.

Exceptions:

1. If selected value for the property is erroneous, system rejects the request and displays an error message.
2. Selected value for the property can conflict with another preference, at this situation a warning message is displayed and user is asked for authorization.

Use Case: Change Toolbar Preferences

Primary Actor: Web Application Developer
Goal in Context: Enable user to add/remove or change orientation components of toolbar panels.

Scenario:

1. When Web Application Developer executes "Change Toolbar Preferences" process by clicking button in the menu bar.
2. "Change Toolbar Preferences" process displays all possible components of the toolbar.
3. Web Application Developer selects the component to add or remove from the component list.
4. Web Application Developer confirms its selection by clicking confirm button.
5. System changes toolbars' configuration and display the new version of the toolbar

Exceptions:

1. Selected language file cannot be recognized by the system. System cancels the request and displays an appropriate error message.

4.1.3. Get Preview

Use Case: Get Preview

Primary Actor: Web Application Developer

Goal in Context: To enable Web Application Developer test his/her application in integrated web browser: Mozilla, Firefox

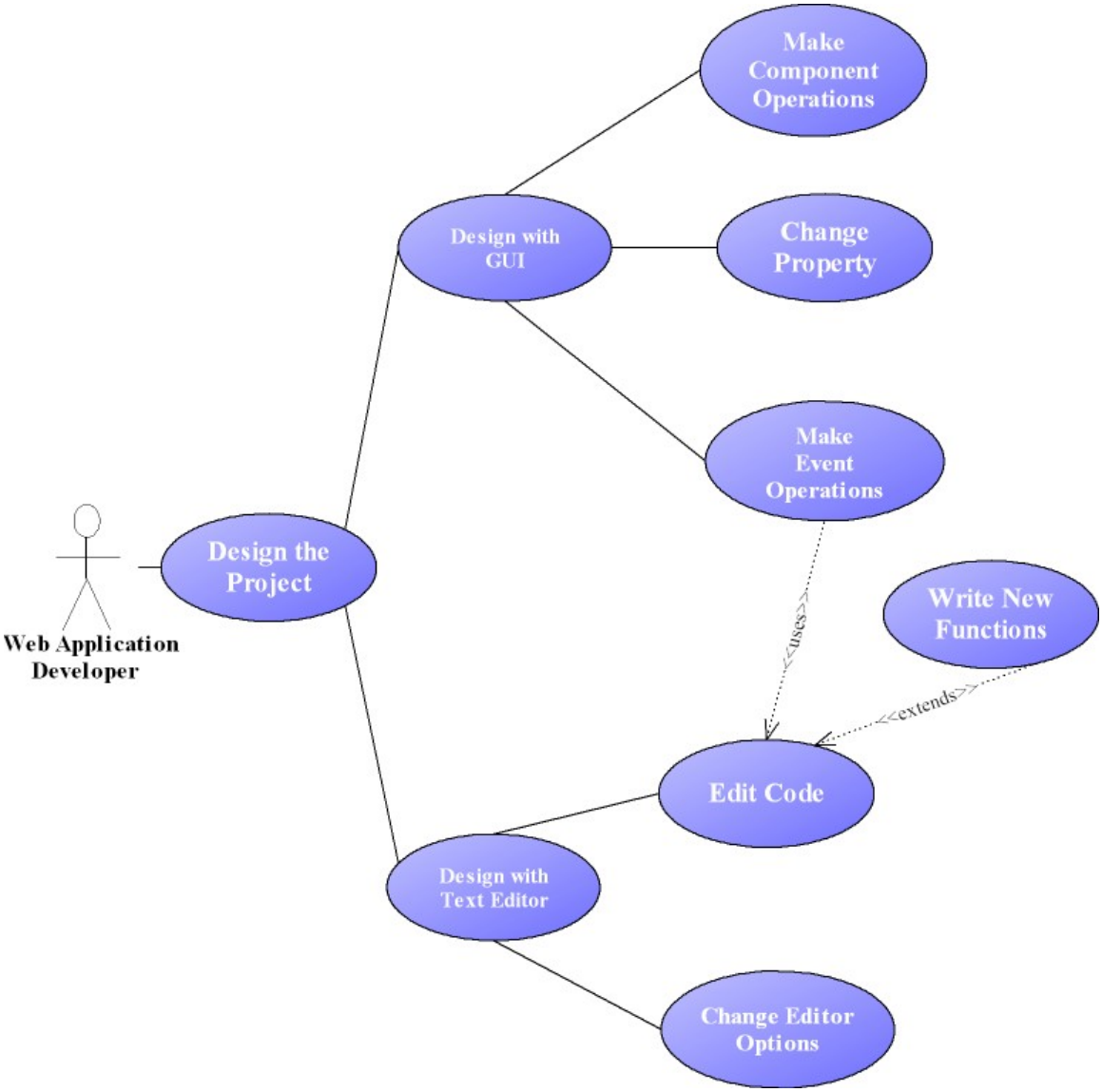
Scenario:

1. Web Application Developer creates web application by using graphical or textural development environment.
2. Web Application Developer change view mode to preview.
3. System changes display mode to preview mode.
4. System displays integrated Mozilla Firefox preview of supplied web application.

Exceptions:

1. Supplied code can be incompatible to view in web browser – system display parts it can distinguish and throws an appropriate error message.
2. Only static parts of the web application (html objects, not JavaScript, flash or database connections) are displayed.

4.1.4. Design the Project



Use Case: Design the Project:

Primary Actor: Web application developer

Goal in Context: To develop the web application, dragging dropping web components, by adding AJAX events, arranging properties of web components, creating code with text editor.

Preconditions: A new project should be created or an existing project should be opened.

Trigger: During the development process, most of the time the web application developer will be in this functionality such as the project id opened with this and when the user presses visual design tab or text editor tab.

Scenario:

1. The web application developer adds web components.
2. The web application developer adds AJAX events.
3. The web application developer adds code to the project.

Use Case: Design with GUI:

Primary Actor: Web application developer

Goal in Context: To develop the web application, to add web page components to the web application, to specify the properties of selected items, to add AJAX event to the application, to add database connection to the application.

Preconditions: A new project should be created or an existing project should be opened.

Triggers: The web application developer decides to develop web application with the GUI and presses the graphical design tab on the screen.

Scenario:

1. The web application developer presses the graphical design tab.
2. The web application developer drags items from the items toolbar.
3. The web application developer drops the dragged item on the graphical design tab.
4. The system displays the dropped items on the design tab.

5. The web application developer adds database connection to a specified database.
6. The web application developer changes the location of the web page components.
7. The web application developer deletes some of the web components.
8. The web application developer changes the properties of the selected web page components.
9. The web application developer adds AJAX events from event editor.

Use Case: Design with Text Editor:

Primary Actor: Web application developer

Goal in Context: To change the generated code by the program, or to add extra functionalities to the application.

Preconditions: A new project should be created or an existing project should be opened.

Trigger: The web application developer decides to generate code with text editor and clicks the text editor tab.

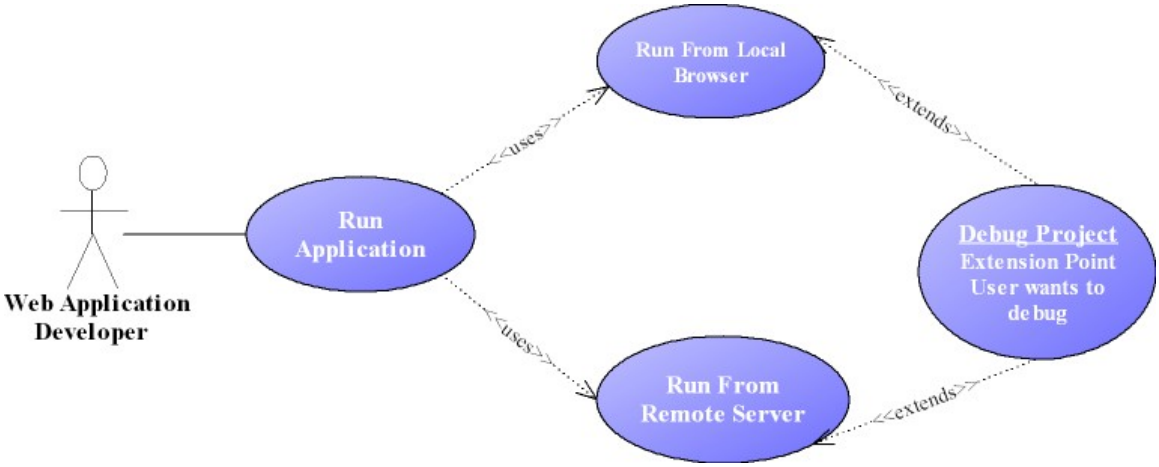
Scenario:

1. The web application developer presses the text editor tab.
2. The web application developer changes some of the text editor options.
3. The web application developer adds functions to this area.
4. The web application developer changes some parts of the previously created functions.

Exceptions:

1. The web application developer is allowed to change not all parts of the system generated code, but only some parts of the system developed code and the functions developer by the web application developer. When the user tries to change the forbidden parts the system inform the user about the situation via a warning message.
2. The web application developer changes the Editor Options. See use-case: "Change Preferences".

4.1.5. Run the Project



Use Case: Run the Project

Primary Actor: Web application developer

Goal in Context: To run the created web application program

Preconditions: All the necessary files should be deployed to the local server or to the remote server. And one of the Mozilla or Internet Explorer browsers should be installed on the local computer.

Trigger: The web application developer decides to run the developed application.

Scenario:

1. The web application developer presses the run button to run the program locally, if the user prefers to run the process in remote server then chooses run from http button. So the deployed files at remote server are started to run.
2. The system does the syntax checking.
3. The system establishes connection with the default browser on the local machine.
4. The deployed files are run on the local browser.

Exceptions:

1. When there is a syntax error on the user generated code, the application will not be displayed on the default browser else an error message will be displayed on the screen and the system asks the user if he/she wants to debug the code (see use case debug project).
2. If there is a problem in running the file at remote server the appropriate error messages will be displayed on the screen of the web application developer.

Use Case: Debug Project:

Primary Actor: Web application developer

Goal in Context: To debug the files of an existing project.

Preconditions: There should be an open project to debug.

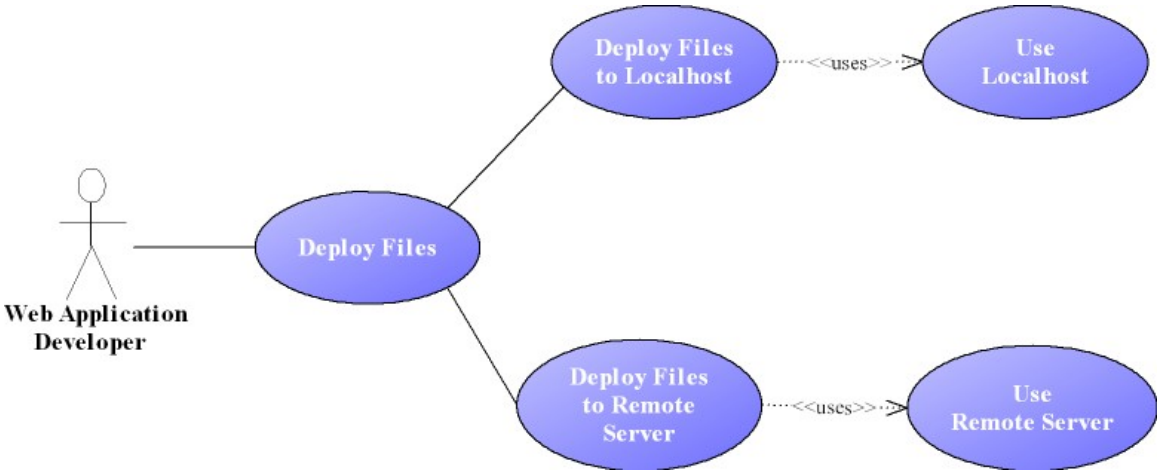
Trigger: If an error occurs just before run process and the user wants to debug the files. Or the user clicks the debug button on the menu bar.

Scenario:

1. The user clicks the debug button on the menu bar.
2. The syntax errors on the text editor part are displayed to the user by the system.
3. The variables of the code will be displayed on the debug panel.

4. The user corrects the parsing errors.
5. The user again presses the debug button.
6. The system informs the user about successful debugging process.

4.1.6. Deploying the Project



Use Case: Deploy Files

Primary Actor: Web Application Developer

Goal in Context: To enable Web Application Developer to put and save project files in appropriate and desired locations at local or remote machine. Remote machine uploads will be done via FTP.

Preconditions: FTP adjustments should be satisfied properly.

Trigger: Web Application Developer clicks “Save File” or “Upload File” button in menu bar or pressing keyboard shortcut to trigger “Deploy Files” process.

Scenario:

1. Web Application Developer creates web application by using graphical or textural

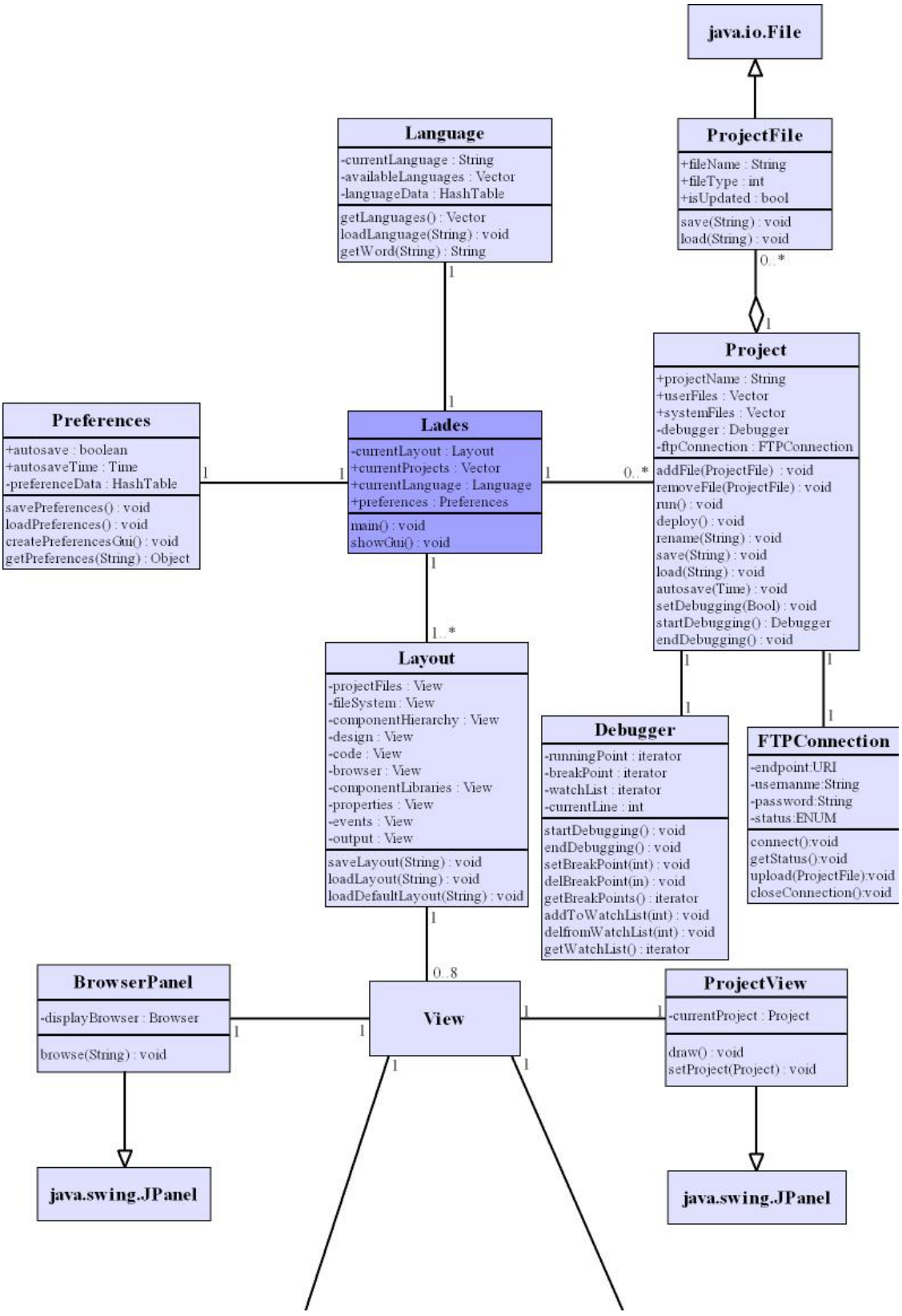
development environment.

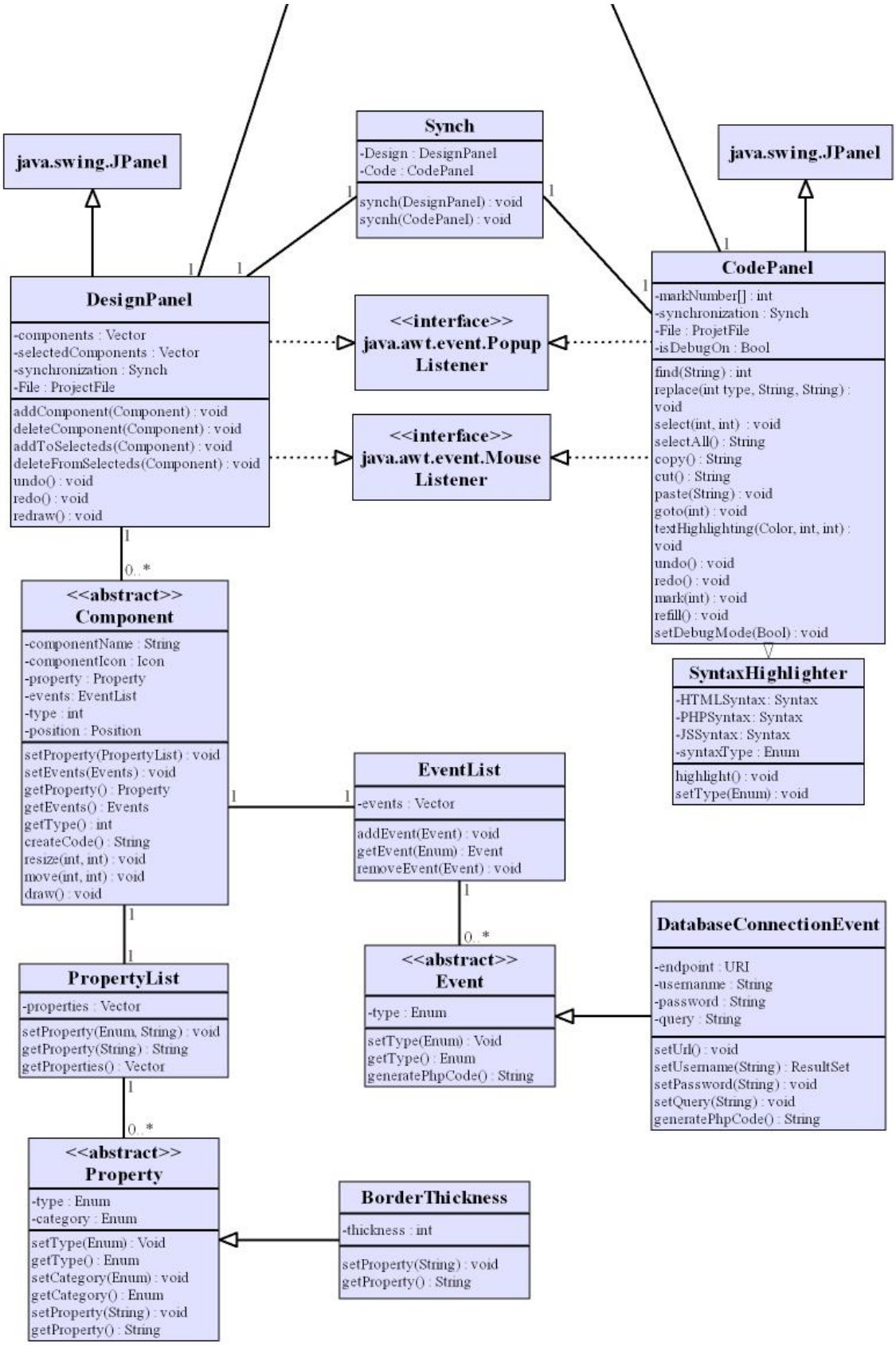
2. Web Application Developer selects save or upload file property by clicking appropriate button or pressing keyboard functions.
3. System displays default location to save or upload file. Web Application Developer can change the location to save or upload file and confirms the location.
4. If save to local machine option is selected, system saves the file to local machine's given location.
5. If user select upload to remote machine option, system creates a FTP connection to that remote machine. After establishing successful FTP connection to remote machine, system uploads file to selected location.
6. System displays successful or error message to Web Application Developer.

Exceptions:

1. User does not have necessary rights for saving file – file cannot be saved to given location and an error message is displayed.
2. FTP connection cannot be established with remote machine – an error message indicating connection problem is displayed.
3. During file transfer with via FTP connection, connection can be lost due to external problems like Ethernet cable can be unplugged. In this case file upload cancels and an error message is displayed.
4. Web Application Developer can enter wrong file names – like containing "" character or prep reserved file names like “con” in MS Windows –. In this situation system asks Web Application Developer to enter a correct file name.

4.2. Class Diagrams




java.swing.JPanel
Synch
-Design : DesignPanel
-Code : CodePanel
synch(DesignPanel) : void
sycnh(CodePanel) : void
java.swing.JPanel
DesignPanel
-components : Vector
-selectedComponents : Vector
-synchronization : Synch
-File : ProjectFile
addComponent(Component) : void
deleteComponent(Component) : void
addToSelecteds(Component) : void
deleteFromSelecteds(Component) : void
undo() : void
redo() : void
redraw() : void
<<interface>>
java.awt.event.Popup
Listener
<<interface>>
java.awt.event.Mouse
Listener
CodePanel
-markNumber[] : int
-synchronization : Synch
-File : ProjetFile
-isDebugOn : Bool
find(String) : int
replace(int type, String, String) :
void
select(int, int) : void
selectAll() : String
copy() : String
cut() : String
paste(String) : void
goto(int) : void
textHighlighting(Color, int, int) :
void
undo() : void
redo() : void
mark(int) : void
refill() : void
setDebugMode(Bool) : void
SyntaxHighlighter
-HTMLSyntax: Syntax
-PHPSyntax: Syntax
-JSSyntax: Syntax
-syntaxType : Enum
highlight() : void
setType(Enum) : void
<<abstract>>
Component
-componentName : String
-componentIcon : Icon
-property : Property
-events: EventList
-type : int
-position : Position
setProperty(PropertyList) : void
setEvents(Events) : void
getProperty() : Property
getEvents() : Events
getType() : int
createCode() : String
resize(int, int) : void
move(int, int) : void
draw() : void
EventList
-events : Vector
addEvent(Event) : void
getEvent(Enum) : Event
removeEvent(Event) : void
DatabaseConnectionEvent
-endpoint : URI
-username : String
-password : String
-query : String
setUrl() : void
setUsername(String) : ResultSet
setPassword(String) : void
setQuery(String) : void
generatePhpCode() : String
<<abstract>>
Event
-type : Enum
setType(Enum) : Void
getType() : Enum
generatePhpCode() : String
PropertyList
-properties : Vector
setProperty(Enum, String) : void
getProperty(String) : String
getProperties() : Vector
<<abstract>>
Property
-type : Enum
-category : Enum
setType(Enum) : Void
getType() : Enum
setCategory(Enum) : void
getCategory() : Enum
setProperty(String) : void
getProperty() : String
BorderThickness
-thickness : int
setProperty(String) : void
getProperty() : String

These are the classes that will be used in LADES. The public variables begin with '+', the private ones begin with '-'. Since all the methods declared above are public ones, they are not explicitly marked as public. The methods define the communication between classes. Since most of the inner workings of these classes are still not decided, they are not shown. All the classes are explained below:

- **LADES**: This class will have the main function. This class will be the main bridge between the current projects (from the whole project pool) and the main GUI. It will be responsible for managing *Language* class and the *Preferences* class. The intent of making three of theses classes is to make them available to other classes without complex methods.

- **Language**: This class will read language files from a predefined directory to get available languages. Other classes will probe this class in order to retrieve the required words used in the whole GUI. It will be created by *LADES* but after creation other classes will be able to access this class with *LADES.currentLanguage*.

- **Preferences:** Lades preferences will be saved in a file named "Lades.xml" by Preferences class. It is also responsible for displaying preferences window when requested. Other classes will be able to query the current value of an option. It will be created by *LADES* but after creation other classes will be able to access this class with *LADES.preferences*.

- **Project:** This class will respond to a project created by Lades. It will have options for managing a project; such as adding/removing a file to/from a project, running/deploying a project, saving/loading a project etc. It will have a list of files associated to it and will have an *FTPConnection* for deploying

purposes. Each project will have its own *Debugger* class for debugging purposes. Other classes will invoke *Project's startDebugging* and *endDebugging* functions to use this feature. All other methods of will be called from *Debugger* directly. It will be created by *LADES* but after creation other classes will be able to access this class with *LADES.currentProjectss*.

- **Debugger:** This class stands responsible for debugging purposes. Debugging of JavaScript files will be handled *Debugger*. The users of this class will be able to set break points and watch variables. In order to use this class one must call the associated *Project* class first. After initialization its methods can be used directly.

- **ProjectFile:** The files associated to a project will be represented by this class. It will have save and load methods and will be responsible for deciding if a file is updated or not. It will inherit java.io.File to use its methods. This class will be accessible via its associated *Project* class.

- **FTPConnection:** This will be used *ProjectFile* to establish and use FTP connections. GUI classes will active FTP connection indirectly via *Project.*

- **Layout:** This class will be created by *LADES* when showgui() method is invoked. It will be responsible for initiating views and saving/loading whole GUI layout.

- **View:** This is a class taken from an open source library called InfoNode. This class composes a JPanel object. It will enhance

its JPanel object with docking, undocking, tabbing, minimizing, maximizing etc. Our own View objects will compose their own related panel objects that inherits JPanel object.

- **BrowserPanel:** This class will display the current project file via built-in browser. It will inherit java.swing.JPanel.

- **ProjectView** This class will display the files in the current project. It will inherit java.swing.JPanel.

- **Synch:** *Synch* will be the main bridge between *CodePanel* and *ViewPanel*. Both of these mentioned classes will compose a *Synch* class. This class will have only two methods: one for converting the components displayed in *CodePanel* to HTML code for *ViewPanel* and the other one for the reverse of the former, converting code to components.

- **CodePanel:** This panel will have an embedded editor. It will provide the code viewing of the *ProjectFile* showed in *DesignPanel*. This editor will have many useful features such as: copy, paste, cut, find & replace, undo, redo, line marking, syntax highlighting etc. Syntax highlighting ability will be provided by *SyntaxHighlighter* class. This class will have java.awt.event.PopupListener and java.awt.event.MouseListener interfaces. java.swing.JPanel will be inherited by *CodePanel*.

- **SyntaxHighlighter:** The syntax highlighting feature of *CodePanel* will be provided by this class. Syntax highlighting will work for recognized file

extensions such as HTML, JS (JavaScript), PHP (PHP Hypertext Preprocessor), CSS (cascading style sheet).

- **DesignPanel:** This Panel will be responsible for handling visual creation of *ProjectFile*s. It will provide the visual viewing of the file showed in *CodePanel*. Main user friendly features, for creating files, of LADES project will be implemented in this class such as: drag & drop adding of *Component*s (these will be explained later), event creation, event handling wizards etc. *DesignPanel* class will implement interfaces of java.awt.event.PopupListener and java.awt.event.MouseListener. java.swing.JPanel will be inherited.

- **Component:** The components that can be added via GUI of *DesignPanel* are represented by *Component* class. This class is an abstract class. Specific Components will inherit this class to use it as a base class. Some of these components are: tables, forms, buttons; specifically HTML objects. As there are a lot of HTML components that will inherit *Component*, those classes are not shown, only the abstract class is shown to visualize the general interaction between classes. Each *Component* will have its HTML properties stored in *PropertyList* and associated events stored in *EventList*.

- **PropertyList:** This class will hold information about the properties of HTML objects such as border, height, color etc. It will store a vector of specific *Property* elements. *Component* will use this class's methods to add, delete or change HTML properties.

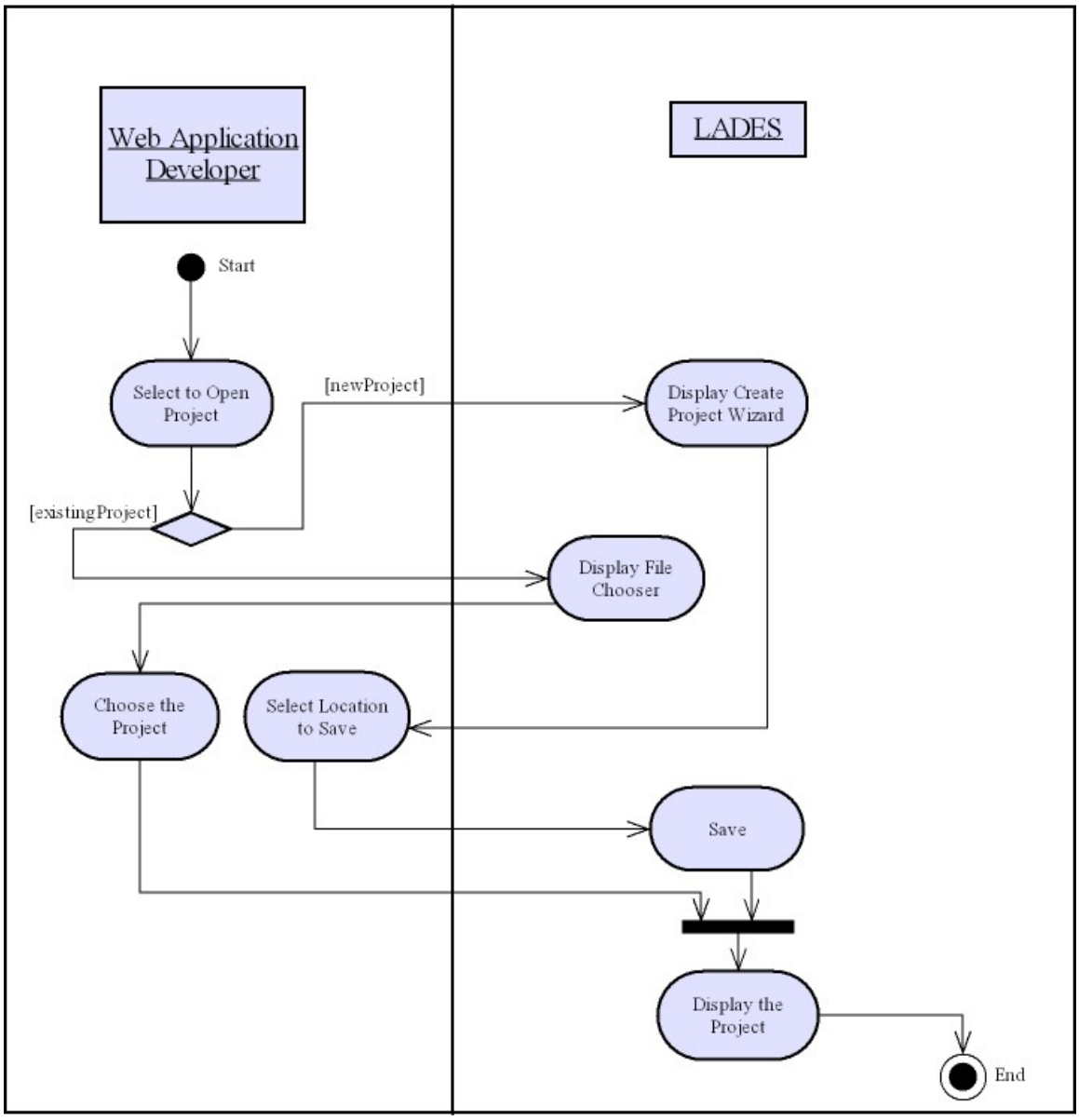
- **Property:** This abstract class will be the base for storing HTML elements properties. Some of these properties are: border thickness, height, color etc. All of these properties will inherit this class and use it as a base class. *BorderThickness* class has been added as an example inheriting *Property* class.

- **EventList:** This class will hold information about the events, and their actions, associated to a specific HTML component. Some of these events are: onhover, onclick, ondblclick etc. These events are JavaScript events that will be tied to a JavaScript function. For example a HTML "submit button" may use its onhover trigger to run a function named "getNameList" to get the list of names from a database.

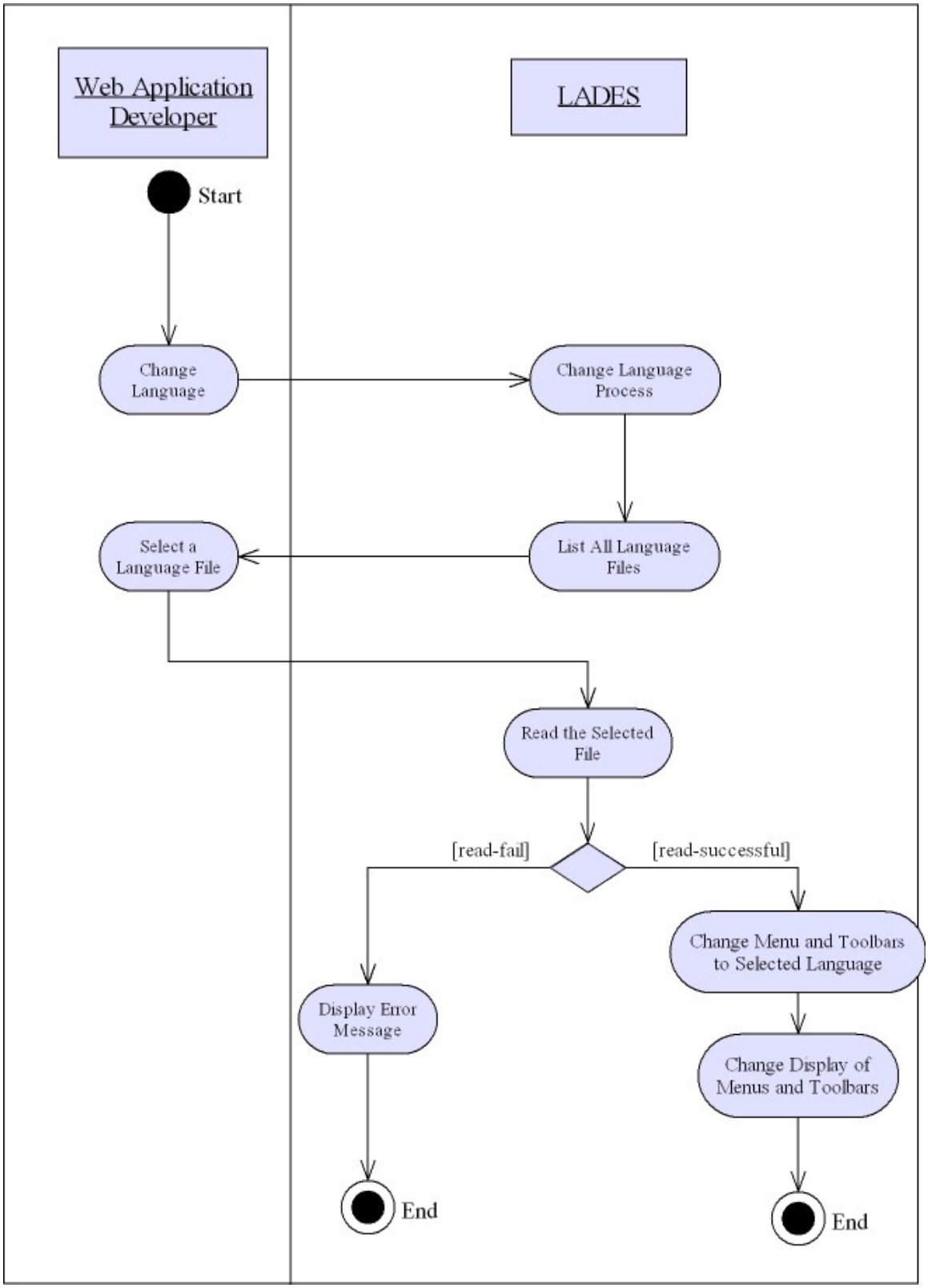
- **Event:** This abstract class will hold information for specific events and their specific actions that can be stored in *EventList* class. The details of the events are explained above in *EventList*. This class will be inherited by all the events that can be generated by JavaScript. *DatabaseConnectionEvent* class has been added as an example inheriting *Events* class.

4.3. Activity Diagrams

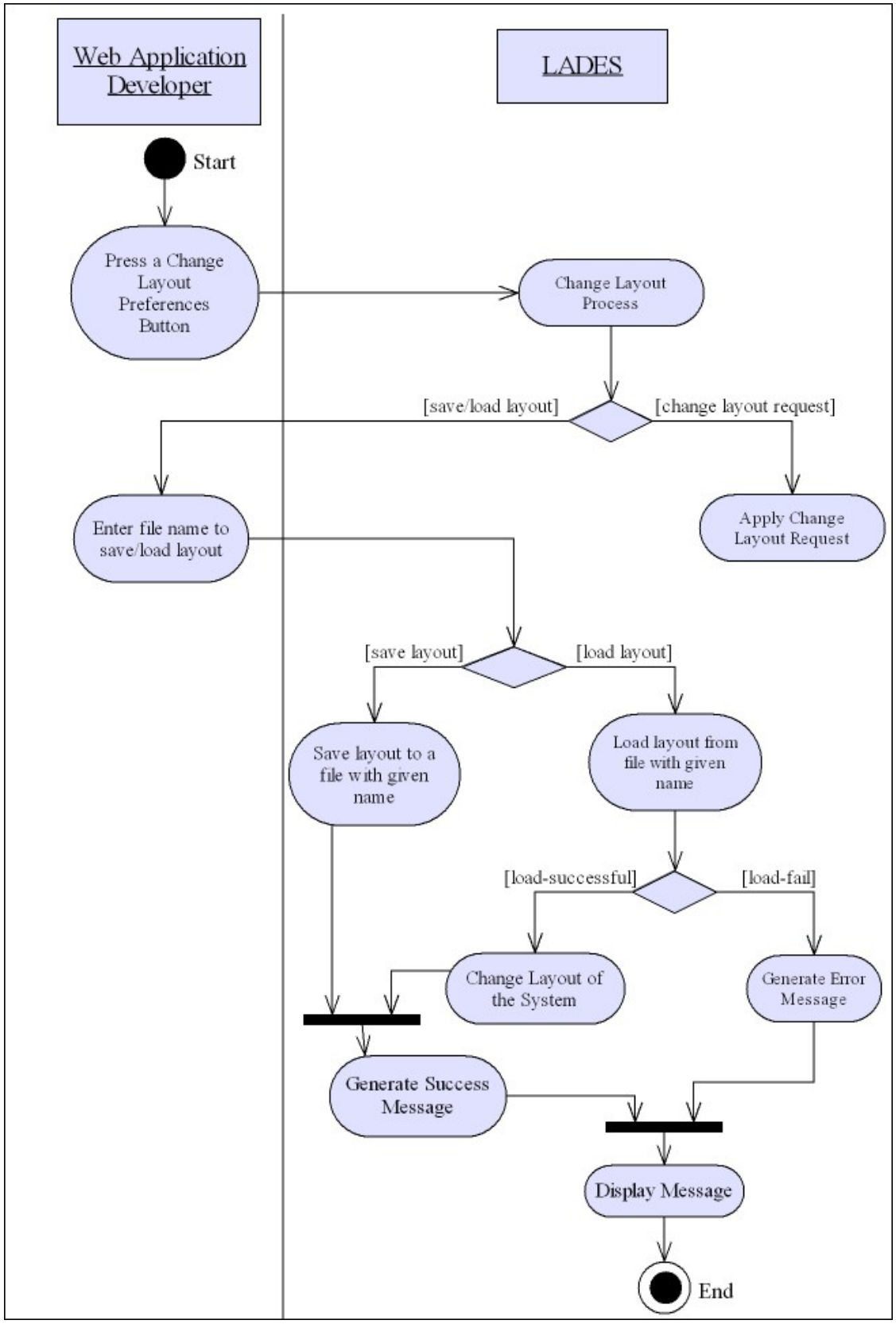
4.3.1. Open Project



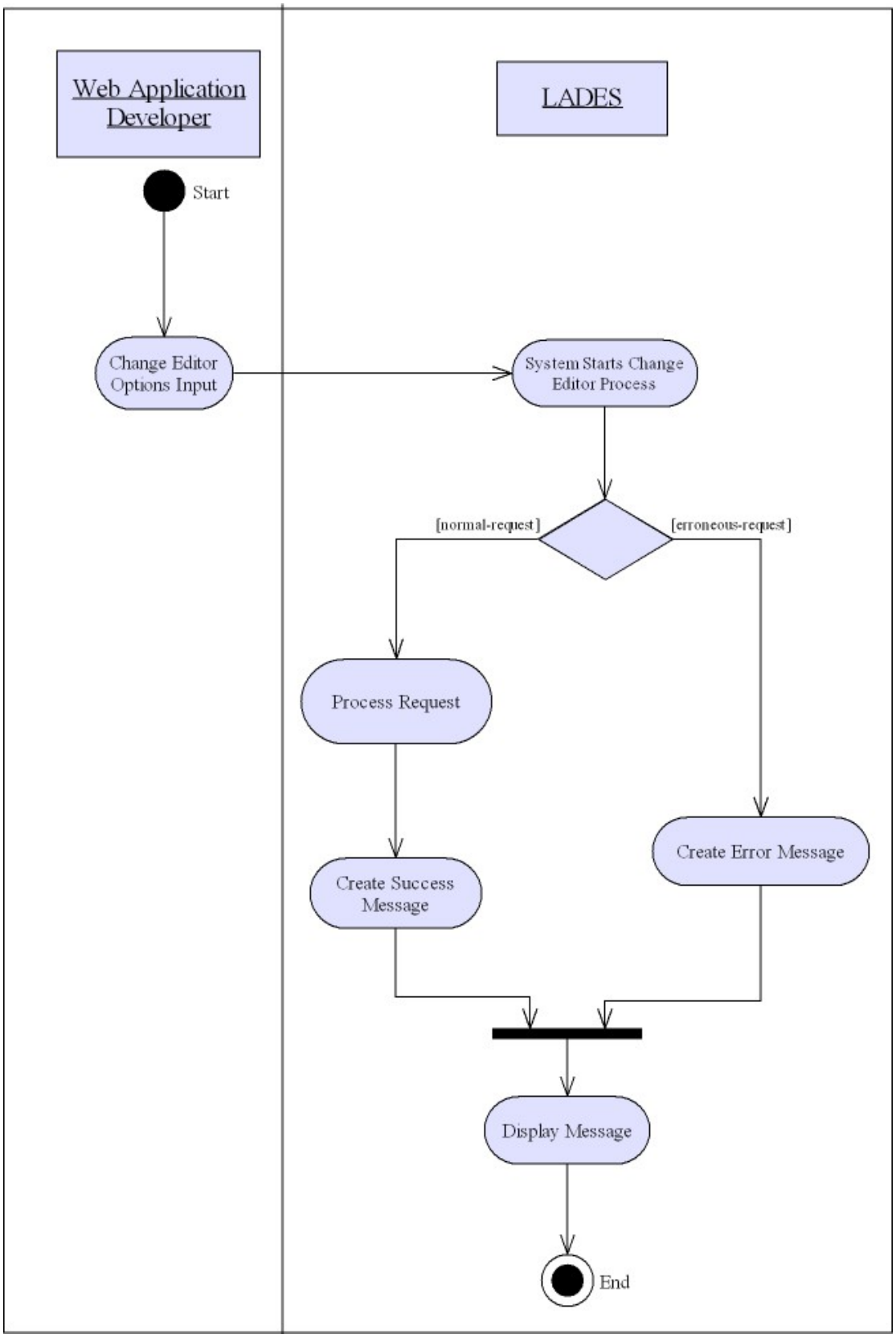
4.3.2. Changing Language



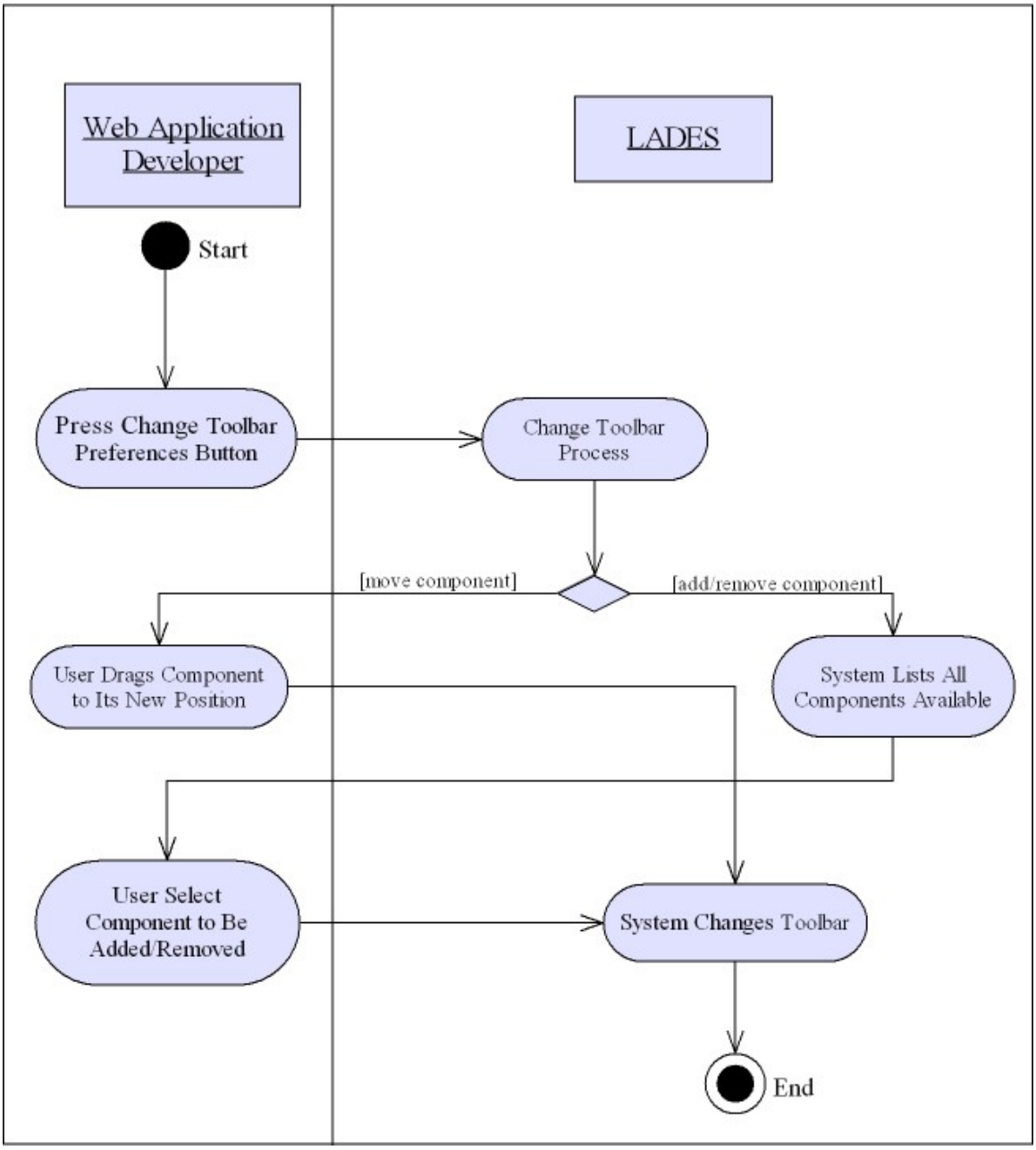
4.3.3. Changing Layout



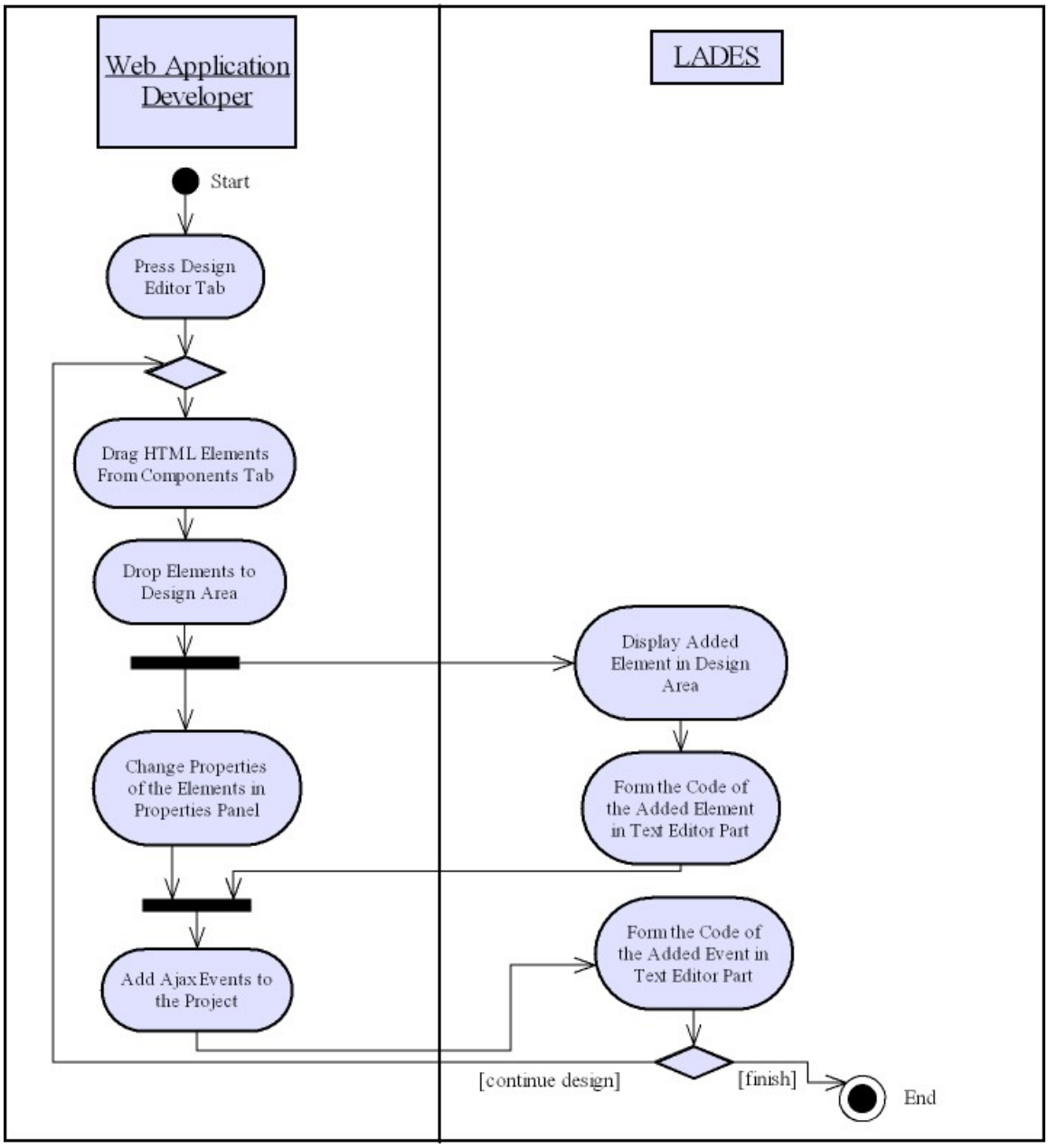
4.3.4. Changing Editor Preferences



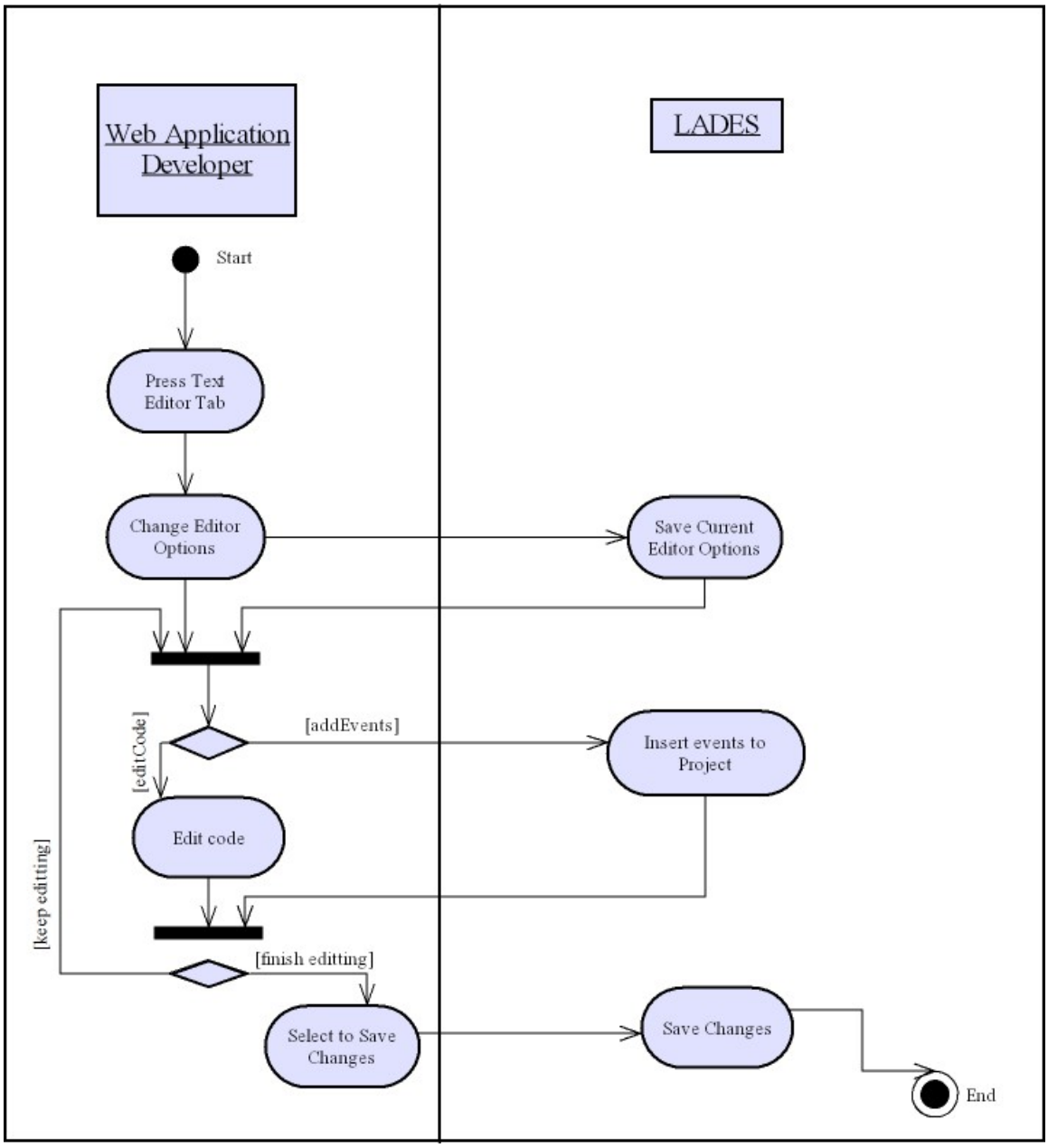
4.3.5. Customizing Toolbar



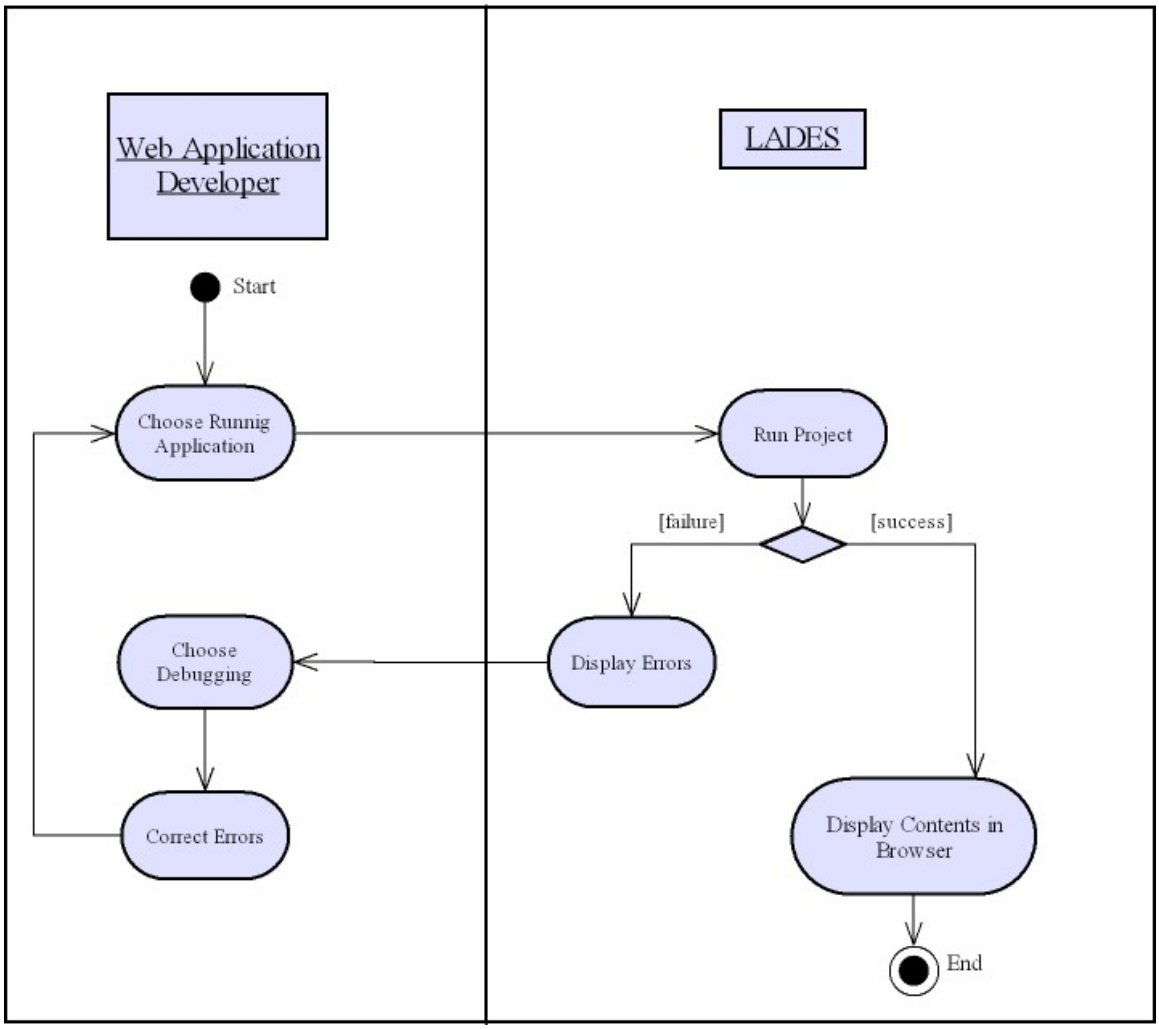
4.3.6. Design the Project Using Design Panel



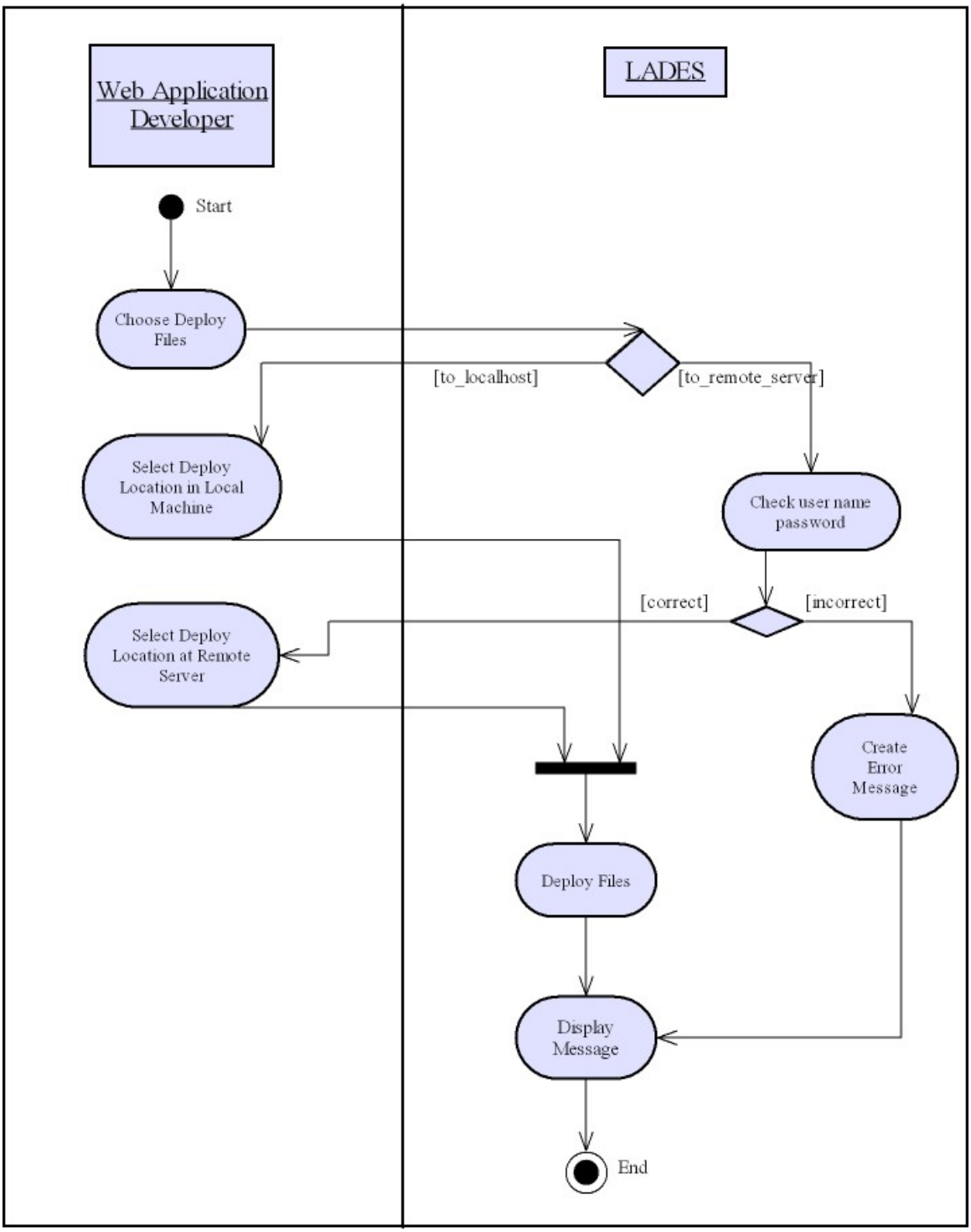
4.3.7. Design the Project Using Code Editor



4.3.8. Run the Project

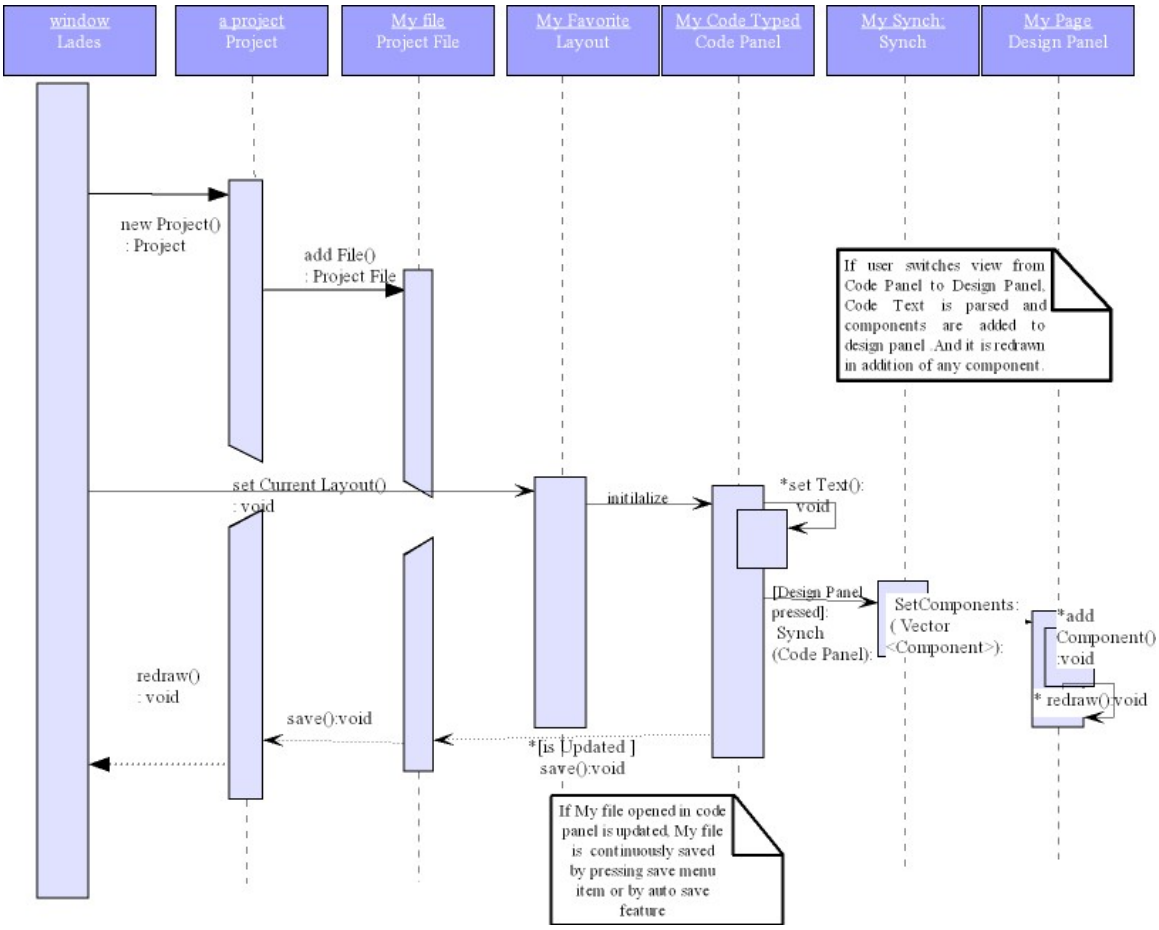


4.3.9. Deploy the Project

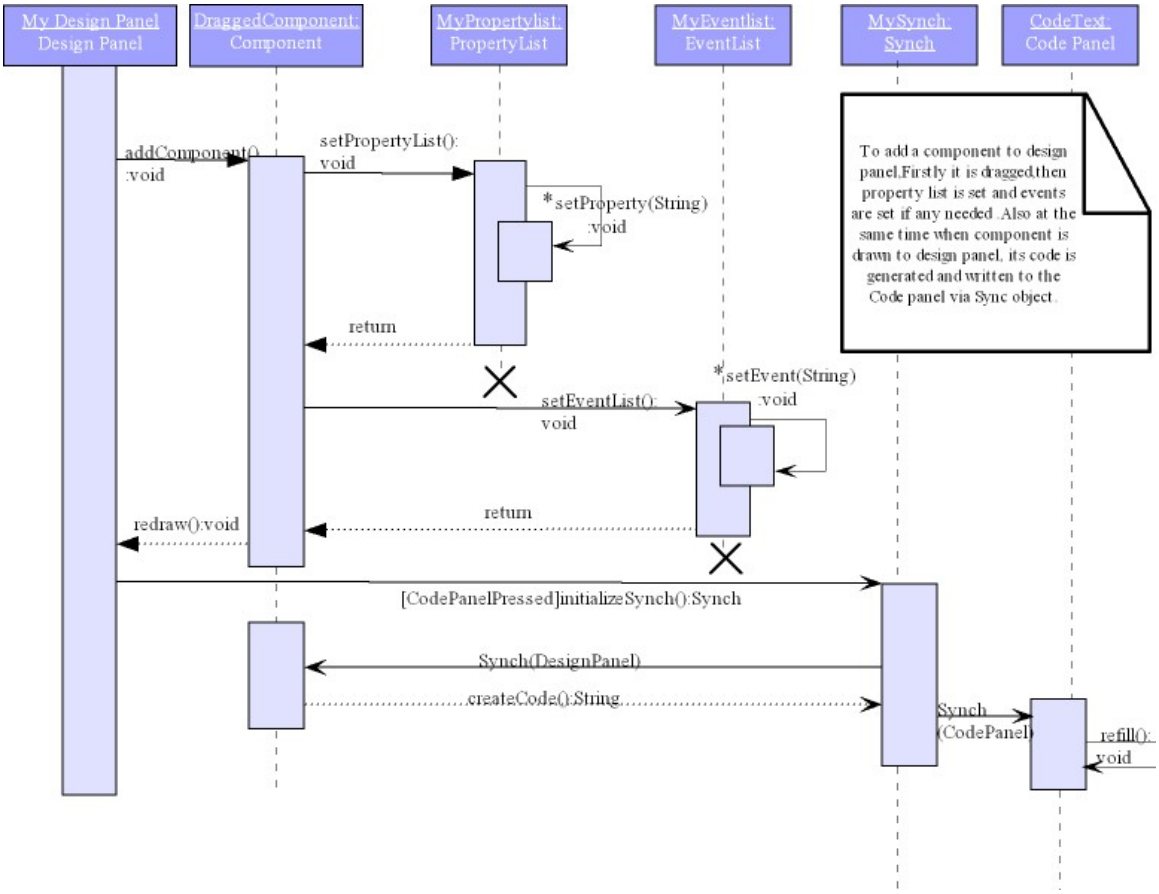


4.4. Sequence Diagrams

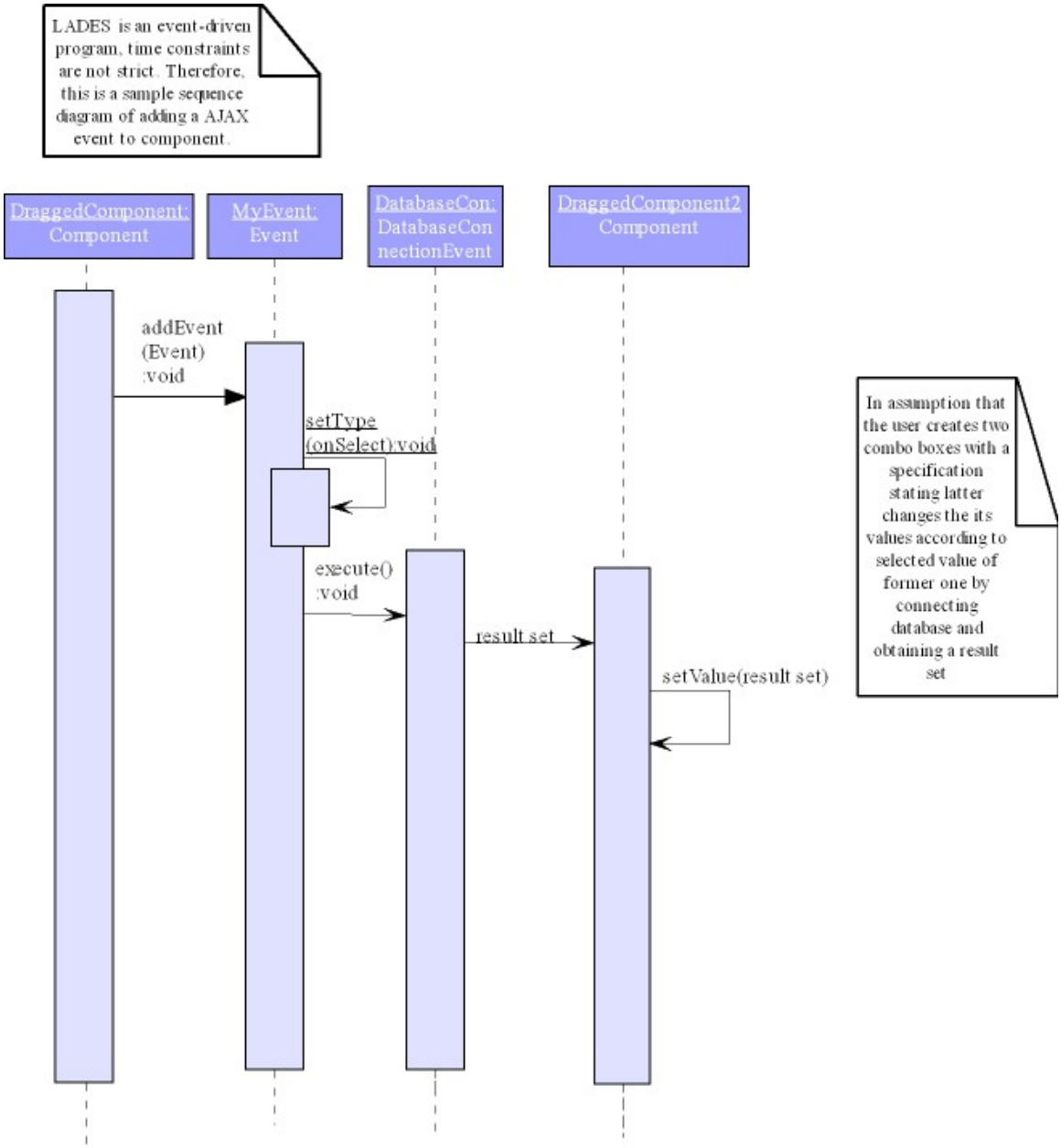
4.4.1. Generating Web Application Page via Code Editor



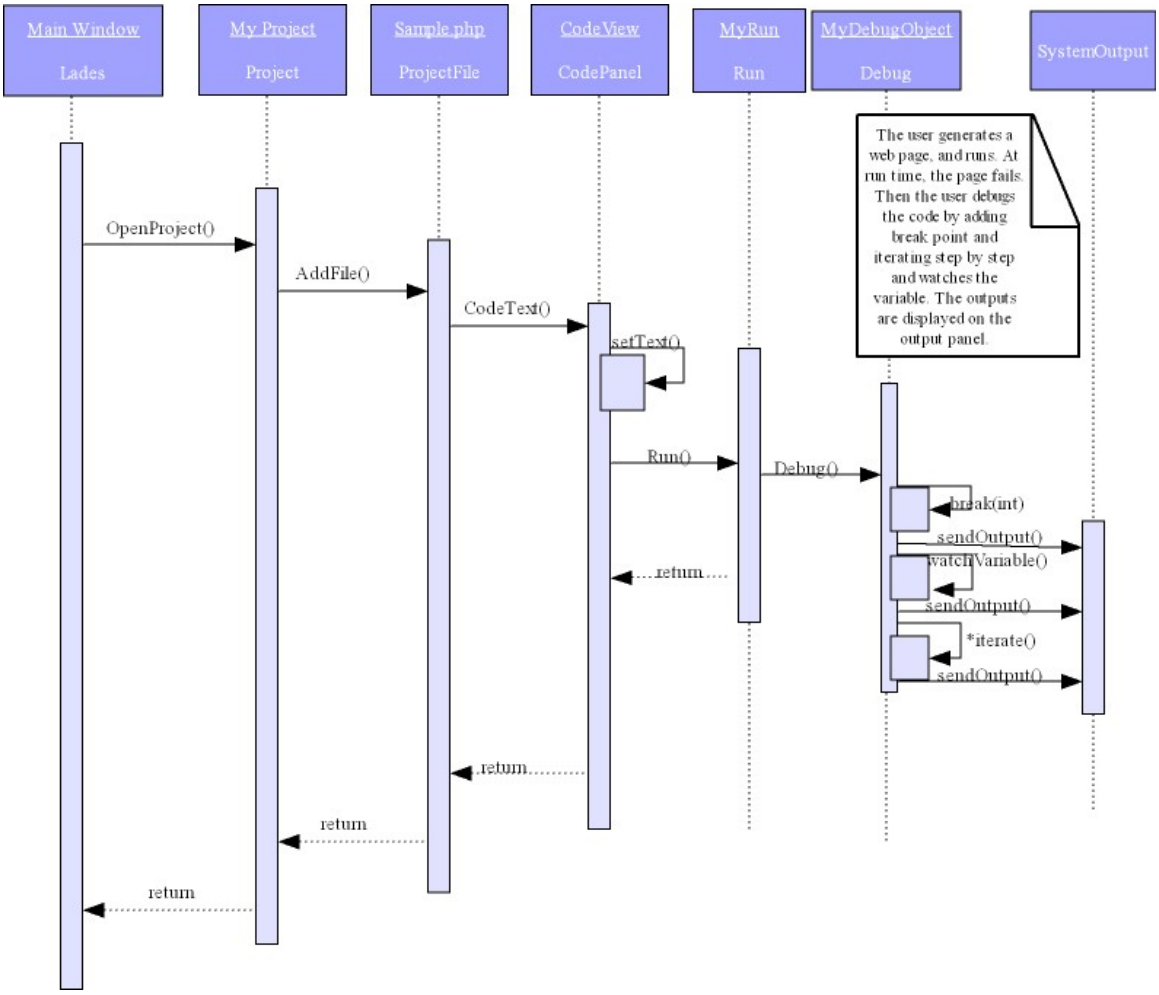
4.4.2. Generating Web Application Page via Design Editor



4.4.3. Adding AJAX Events



4.4.4. Debugging the Project



5. Syntax Definition

In order to achieve flexibility of components, we have stored the components in their own XML files. This approach provides Lades to add new components at a later time easily. In addition to this, user defines components can also be saved with their own XML files.

5.1. Definition of Components

An example of a component, we are providing the XML definition of Text HTML element.

```
<component>
        <name>text</name>
        <icon>img/Label.png</icon>
        <description>text.description</description>
        <tag>span</tag>

        <propertylist>
                <general-properties>
                        <value/>
                </general-properties>

                <text-properties>
                        <color/>
                        <direction/>
                        <letter-spacing/>
                        <text-decoration/>
                        <text-indent/>
                        <text-transform/>
                        <white-space/>
                        <word-spacing/>
                </text-properties>

                <font-properties>
                        <font-style/>
                        <font-variant/>
                        <font-weight/>
                        <font-size/>
                        <font-family/>
                </font-properties>

                <border-properties>
                        <border-width/>
                        <border-style/>
                        <border-color/>
                </border-properties>

                <margin-properties>
                        <margin-top/>
                        <margin-right/>
                        <margin-bottom/>
                        <margin-left/>
                </margin-properties>

                <padding-properties>
                        <padding-top/>
```

```
                <padding-right/>
                <padding-bottom/>
                <padding-left/>
            </padding-properties>

            <background-properties>
                <background-color/>
                <background-image/>
                <background-repeat/>
                <background-attachment/>
                <background-position/>
            </background-properties>

            <position-properties>
                <position/>
                <top/>
                <right/>
                <bottom/>
                <left/>
                <text-align/>
                <vertical-align/>
                <z-index/>
                <clip/>
                <overflow/>
            </position-properties>
        </propertylist>

        <eventlist>
            <onclick/>
            <ondblclick/>
            <onfocus/>
            <onblur/>
            <onmousedown/>
            <onmousemove/>
            <onmouseout/>
            <onmouseover/>
            <onmouseup/>
        </eventlist>
</component>
```

5.2. Definition of a Language File

We are providing the XML definition of Turkish language file. All language files will be in that format.

```
<language>
        <locale>
      TR
   </locale>
        <author>
          Ergin EROGLU
   </author>
        <version>
          1.0.0.0
   </version>
        <words>
          <file>Dosya</file>
          <edit>Düzenle</edit>
          <search>Ara</search>
          <view>Görünüm</view>
          <project>Proje</project>
          <help>Yardım</help>
          <component.hierharchy>Bileşen Yapısı</component.hierharchy>
          <design.panel>Dizayn Paneli</design.panel>
          <file.system>Dosya Yapısı</file.system>
          <output>Çıktı</output>
          <code.view>Kod Penceresi</code.view>
          <design.view>Dizayn Penceresi</design.view>
     <browser.view>Tarayıcı Penceresi</browser.view>
          <new>Yeni</new>
          <open>Aç</open>
          <save>Kaydet</save>
          <find>Ara</find>
          <replace>Değiştir</replace>
          <undo>Geri Al</undo>
          <redo>Yinele</redo>
```

</words>
</language>

6. User Interface

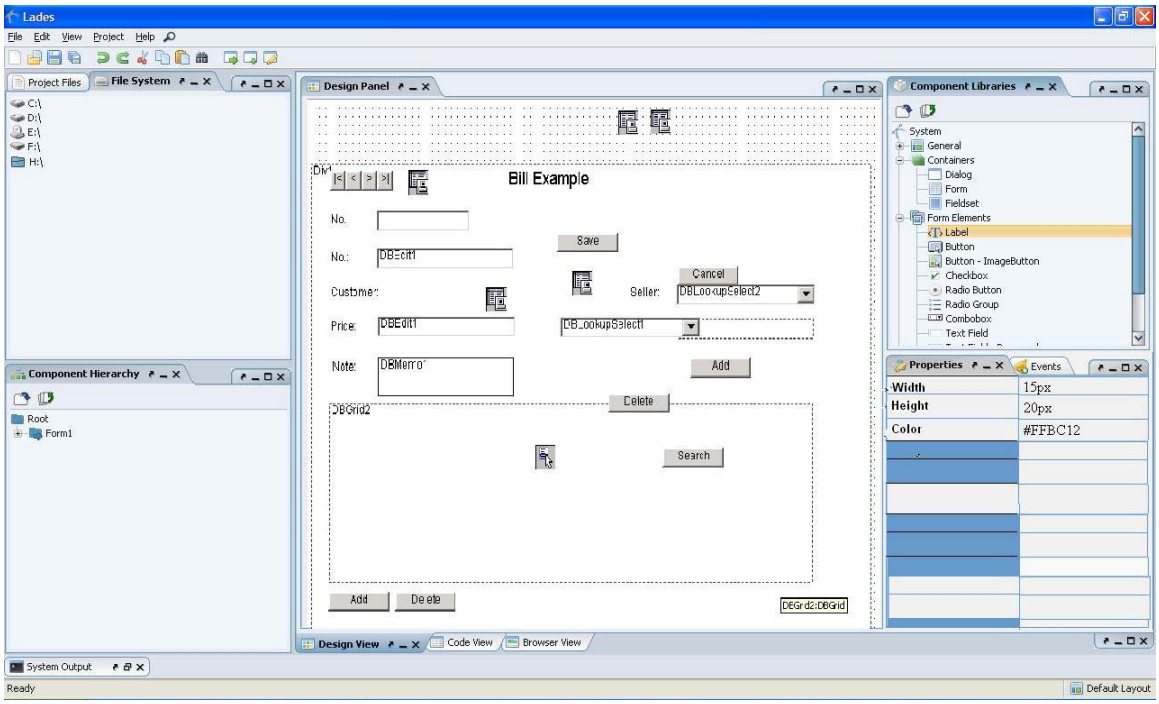


Figure 6 - General overview of Lades

6.1. Layout Design

The user can customize the layout of LADES to his/her preferences. The user can minimize, maximize, close, undock, dock and move around the windows to create a personalized window layout. Also he/she can even take advantage of multiple screens by undocking windows and move them to other screens. When the user is satisfied with the layout it can easily be saved and restored for example the next time the application is started.

In our design interface, total freedom to customize the IDE is given to the user by number of facilities.

6.1.1. Tabbed Panels

There are a lot of tabbed window in our design but all of them can be rearranged by the user by drag and drop or docking or undocking.

When the application size decrease and all of the tabs are not visible you have an option to manage and go around tabs by means of the Figure 7.

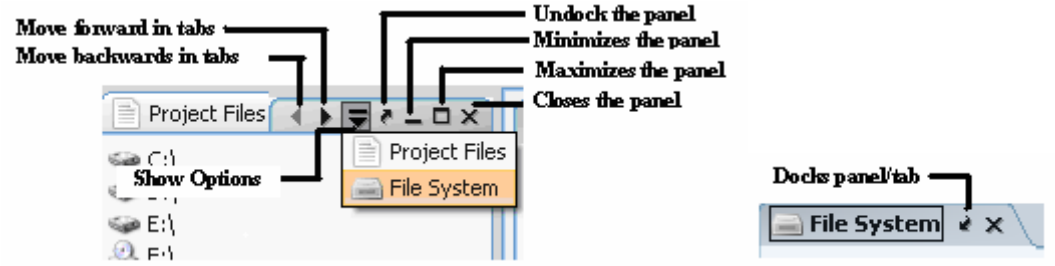


Figure 7 - Tab Manage Options

6.1.2. Panels

Like several Views, their containing panels can also be docked, undocked, minimized, maximized and closed by using their panel manage buttons at the right side of each panel. These buttons are illustrated in Figure 8.

Figure 8 - Panel Manage Options

6.1.3. Customizing Views

In tab windows style user is capable of seeing only one of the tabs at the same time, for going over this handicap; different options are given to the user in LADES to customize the layout. As mentioned above if the user feels more comfortable with the his managed layout these can be saved or set as default to shift between layouts or begin with his/her favorite layout, respectively. There are also options to customize the display.

6.1.3.1. Drag and Drop Views

This drag and drop is performed horizontally within the File Panel which can be seen from Figure 9. The Project File tab is dragged to the bottom of File System tab, by this way, two tabs can be seen at the same time. The same operation can be performed by vertical dropping which splits the tabs into two vertically, and arranges them side by side.

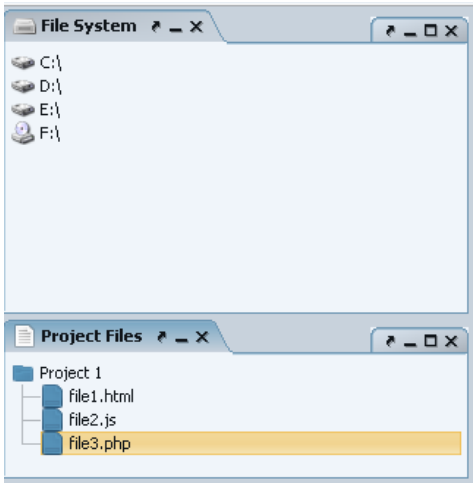


Figure 9 - Drag and Drop Views

6.1.3.2. Undocking Views

All the views available in LADES can be undocked to customize the layout. When a view is undocked, it creates a new window on the fly which can be moved through your entire screen. As an example, shows the undocked view of “Project Files” view.

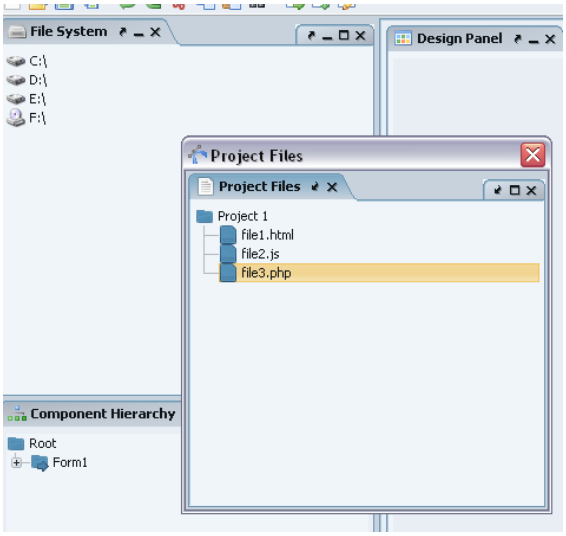


Figure 10 - Undocking Views

6.2. Menu bar

A menu bar is provided by LADES to implement windows. The menu bar allows the user to have point-and-click access to window-specific functions.

6.2.1. File Menu

6.2.1.1. New

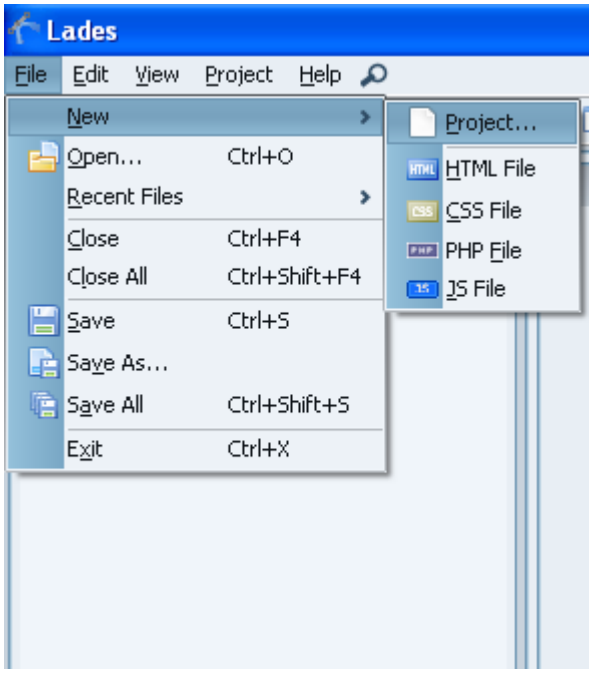


Figure 11 - File Menu

In the File Menu Option of the LADES, user is given different options to create the web application pages.

- User can create New Project by using "Create a New Project Wizard". By the help of the wizard a directory will be created and located according to the user's preferences and after creating a project, new files will also be located under created directory of the project. The user will begin with either an Empty Project or Hello World Example.

- The user can also create just one File by selecting either HTML File or CSS File or PHP File or JS File. In this menu, he/she will also have an option to begin with empty or simple example program.

6.2.1.2. Open

The existing projects or files can be visualized and edited by the selecting open option of file menu. And conventional shortcut, Ctrl + O, for this open menu will also be supplied by the LADES. After selecting this option a JFileChooser will be opened to browse the path of the file.

6.2.1.3. Recent Files

- When the user wants to continue with the project or the file, he/she has recently working on, a useful option is provided by LADES by an option named “Recent Files” In this option recently used files are listed to be accessed easily. The file is opened when the user is clicked on it.

6.2.1.4. Close

- The Close option will close the file on which the cursor is. If there is a change in the code after last save, it shows a warning message and asks for save, cancel or ignore.
- The Close All option will close all the files which are currently in use or opened. This option performs the close action by stated above functionality on all files.

6.2.1.5. Save

- The Save option will save the file on which the cursor is to its own location. And this option can also be performed by the conventional shortcut, Ctrl + S.
- The Save As will save the file on which the cursor is to the specified location by the help of JFileChooser menu.
- The Save All option will save all the files which are currently in use or opened. This option performs the save action by stated above functionality on all files. And conventional shortcut, Ctrl + Shift + S, for this open menu will also be supplied by the LADES.

6.2.1.6. Exit

- Exit performs save all option and safely exit of the LADES.

6.2.2. Edit Menu

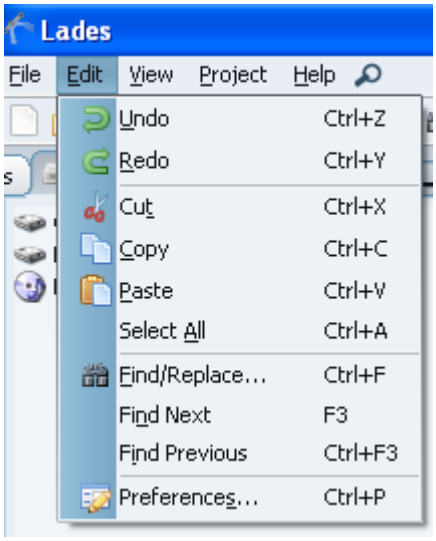


Figure 12 - Edit Menu

6.2.2.1. Undo

This is the one of the most wanted and helpful functionality of a Text Editor should have to save time and effort. It erases the last change done to the document reverting it back to an older state.

6.2.2.2. Redo

Redo is the reverses of the undo action by the help of undo buffer and advances the buffer to a more current state.

Cut, Copy, Paste, Select All, Find/Replace, Find Next, Find Previous provided in LADES are same with conventions.

6.2.2.3. Preferences

Lades preferences can be reached by this menu item. This provides a dialog box to change various settings of the IDE like auto saving, several text editor options…

6.2.3. View Menu

View Menu supplies different options in the layout of the LADES. All View Panels are shown when LADES is opened in its default. Then user will customize these shown views through this panel. When a view is selected a tick image will be shown or when it is deselected the tick will be hidden. And in the layout only the ticked views will be shown to the user. The views can be chosen in any combination.

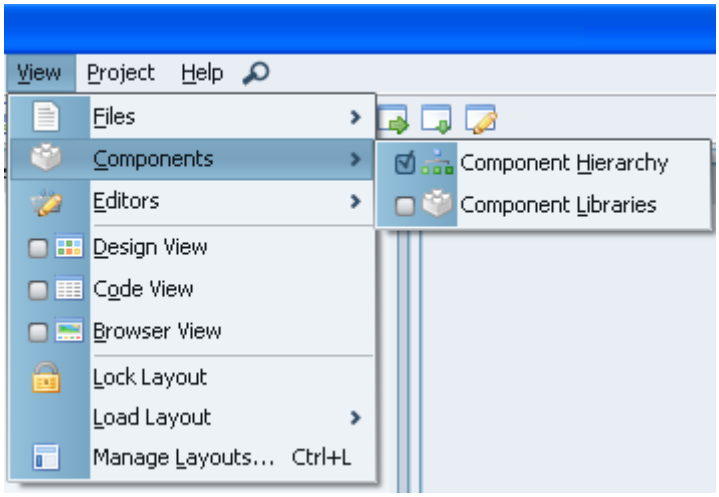


Figure 13 - View Menu

6.2.3.1. Files

LADES will provide two file view option in two consecutive tabs in tree view; File System and Project Files (see). The views can be chosen in any combination.

6.2.3.2. Components

Similar to File view functionality, this provides component Hierarchy and component Library (see Figure XXX- Component Hierarchy and Figure XXX- Component Library) to be visible.

6.2.3.3. Editors

Similar to File view functionality, this provides Events and Properties (see Figure XXX- Properties and Figure XXX- Events) to be visible.

6.2.3.4. Views

Design View, Code View and Browser View used to set the main panel to the selected View Figure XXX –XXX shows the views in detail.

6.2.3.5. Lock Layout

It is used to save layout and keep it stable during the run time of the IDE. Undocking and docking functionalities of panels and tabs are disabled.

6.2.3.6. Load Layout

This menu lists the saved layouts and loads the selected layout on the fly.

6.2.3.7. Manage Layout

This menu item provides a dialog box to manage the layout system. It provides saving, renaming, deleting layouts.

6.2.4. Project Menu

Project Menu is designed for customizing and managing whole project.

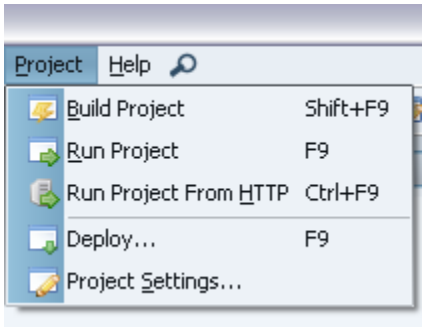


Figure 14 - Project Menu

6.2.4.1. Build Project

The process is also performed through Shift + F9. By building the project syntax checking of the all project files is performed and warning and outputs are displayed to the user via system output panel (Figure XXX).

6.2.4.2. Run Project

Generated web page can be displayed in the default browser by this menu item. Default browser can be selected through the Preferences of Lades. This action runs the project in the local system. F9 key is bound to this action.

6.2.4.3. Run Project from HTTP

This menu item also displays the generated web page in the default browser but it does this action remotely. This means that, generated files are deployed to the remote web server and default browser displays these uploaded files, not the local files. CTRL + F9 is bound to this action.

6.2.4.4. Deploy

To enable Web Application Developer to put and save project files in appropriate and desired locations at local or remote machine. Remote machine uploads will be done via FTP protocol.

6.2.4.5. Project Settings

This option is needed to customize the project setting and to change the default setting according to the user preferences such as display error, syntax highlighting, default save path or deployment setting etc.

6.2.5. Help Menu

Help is the most needed part by user to simplify the usage of the LADES, although Lades is designed as ease usage.

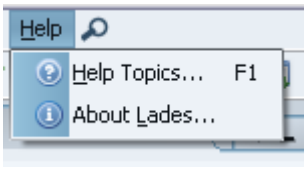


Figure 15 - Help Menu

6.2.5.1. Help Topics

User can use the Help menu to get access to the complete LADES manual and topics which are indexed to ease the search. F1 is a shortcut for Help Topics.

6.2.5.2. About Lades

The menu shows the copyright and version of the LADES and our group name .Also it provides a hyperlink to ACME homepage for further information.

6.2.5.3. Quick Help Search

Search enables the user to search for a word or phrase in the help topics. The text input box seen when the user clicked on the magnifying glass.

Figure 16 - Quick Help Search

6.3. Toolbar

This is a row of icons which performs actions assigned them only with a single click. The assignment is done according to conventions and images given in menu bar to increase the understandability of the action which it performs. Also a description is also displayed when mouse is on it to give a hint to user about its action as seen in Figure 17.

Toolbar can also be customized through the Preferences dialog box. By using this, user has the option to show his/her own toolbar buttons.

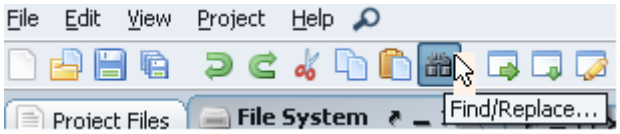


Figure 17 - Toolbar

From left to right, default toolbar button actions can be listed as New Project, Open a Project, Save a File, Save all Files, Undo, Redo, Cut, Copy, Paste, Find and Replace, Run Project, Deploy Project and Project Settings.

6.4. Files Panel

In a web page application project, it is expected to have a number of application files for well-organized working. To display these files to the user and to supply the access to these files by just one click LADES supplies a Project Files Panel which show hierarchy of the files which form the Projects as shown in Figure 18. It is located in left-top side of LADES.

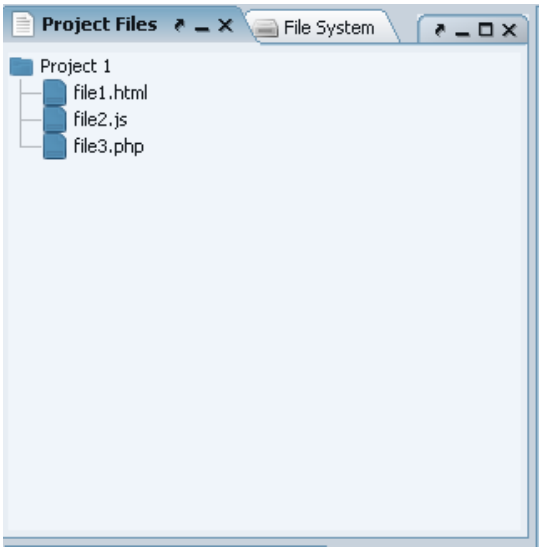


Figure 18 – Project Files

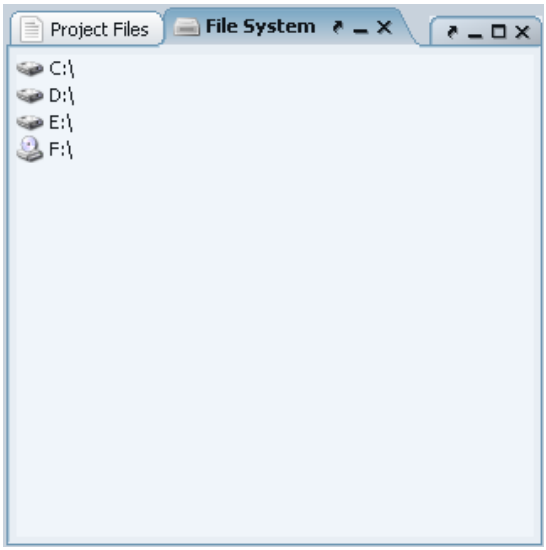


Figure 19 - File System

Also in same panel, there is a File System tab which is used interchangeable by Project Files while via tabs (see Figure 18). In this display, all file hierarchy of machine on which LADES runs is exhibited to the user to locate the project to correct destination.

6.5. Component Hierarchy

The component hierarchy shows the content hierarchy of the web page elements of the web page on which the cursor is. The panel is designed in tree view to show the hierarchy clear and neat. In example, we show that form1 element consists of a Label named with Label1 and a checkbox, Checkbox1, respectively. It is located right-bottom of the LADES which can be seen Figure 20.

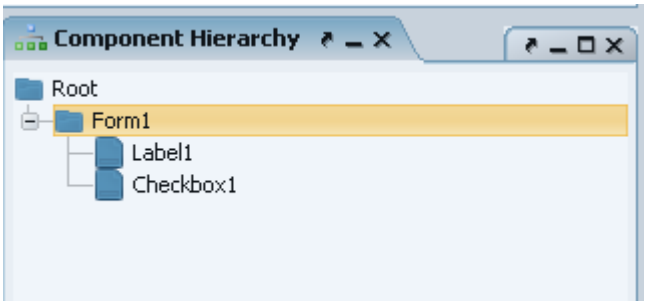


Figure 20 - Component Hierarchy

6.6. Design View

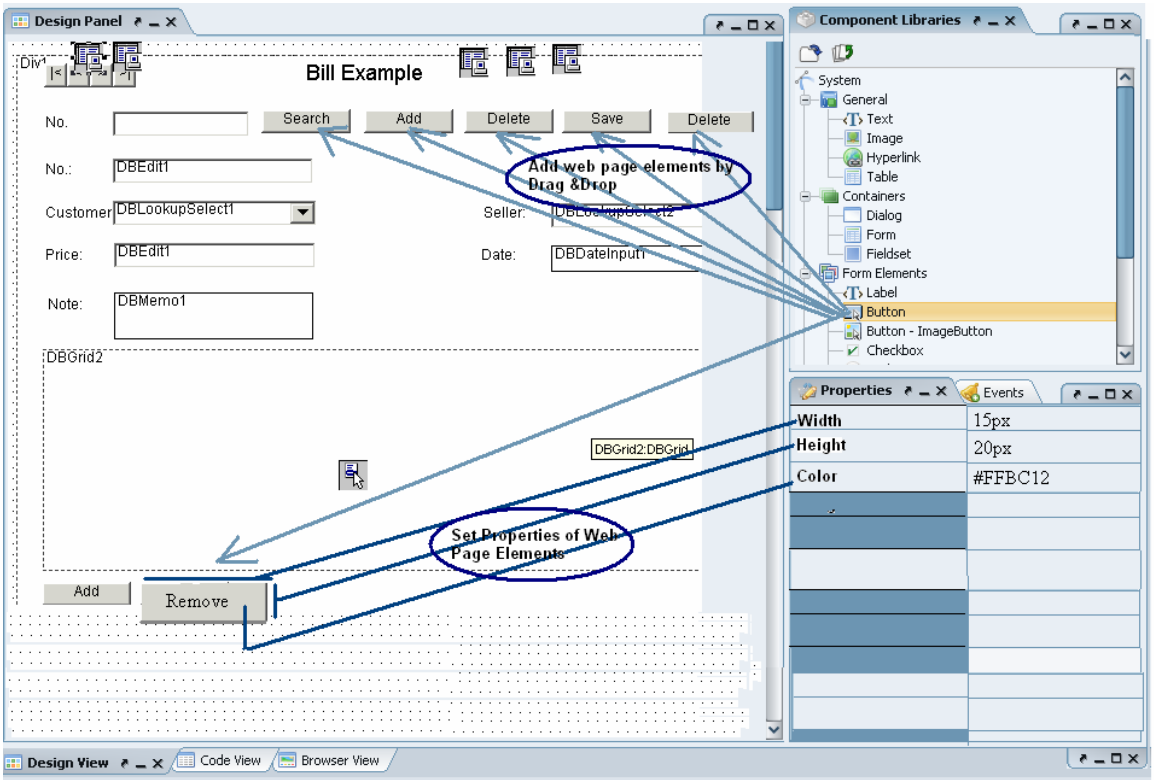


Figure 21 - Design View

Design Panel will allow multiple pages open and will list them in a tabbed panel. User may switch to other files by selecting that file's tab.

6.6.1. Component Libraries

The component library is shown only when the design view is opened in view tabs. Library consists of the web page elements which constructs the web page.

It is a dockable, folding window designed to provide access to graphical user interface components, such as custom graphics, HTML elements. Also an the bottom of this tree view there is a node called User which are the elements defined and customized by the user, and these elements are stored to use latter. It provides simply drag and drop graphic elements and user interface components from the component library onto the Design window as seen in Figure 21.

By drag and drop functionality of listed elements, the design is formed. The component libraries panel is located left-top side of design view window which can be seen Figure 22.

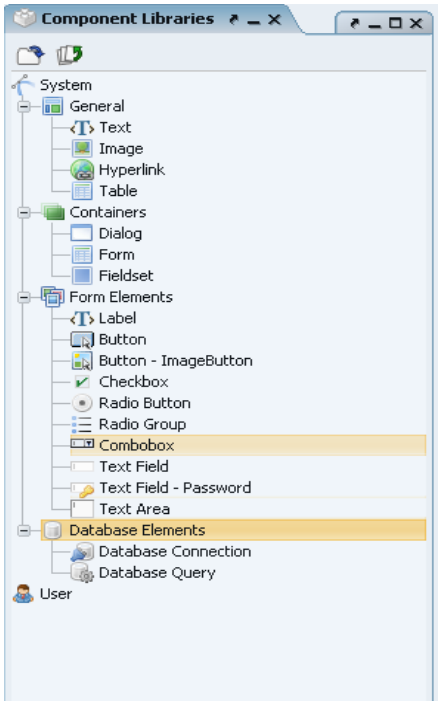


Figure 22 - Component Libraries

6.6.2. Events & Properties Panel

Events and Properties Panel is composed of two tabs, Events and Properties. The tabs of panel are used to arrange and design the structure and functionality of the web page elements, respectively.

The property of the Properties tab can be easily notified from Figure 21, it reshapes the web page form elements. Each component has properties that you can easily edit in the Property Sheets, which are available for standard components and for custom components. Typical properties might include text size and styles, color, or the size of a component.

Events are related with arranging the events like AJAX events etc.

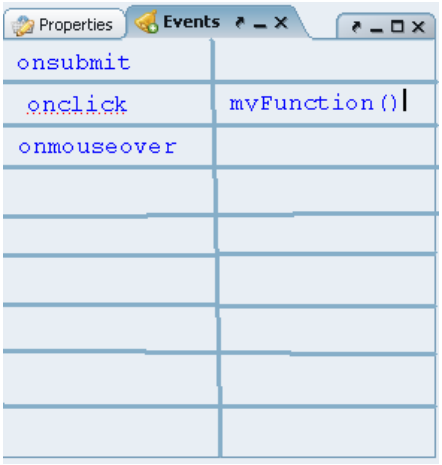


Figure 23 - Events Editor

6.6.3. Design Panel

The user is presented with an empty Canvas, a visual editor in Design Panel. The Canvas provides the work area for laying out the user interface. User can drag and drop graphic elements and user interface components onto the Canvas to visually build applications quickly and intuitively.

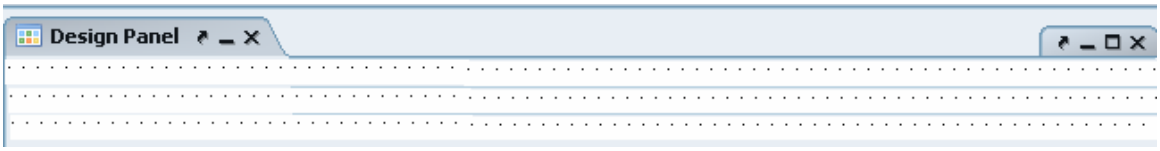


Figure 24 - Design Canvas

6.7. Code View

For those of you who prefer to hand code, or for those who want to see what's happening behind the scenes, LADES provides a Code Editor that supports a number of different required web-based application languages: HTML, JS and PHP. You can move between development modes, from visual to the Code Editor, with ease by help of tabs.

Figure 25 - Code Window

Syntax colorization and formatting for JS, PHP and HTML files, helps the user keep coding issues straight.

6.8. Browser View

The panel shows the user appearance of the generated web page from the aspect of default web browser of the user.

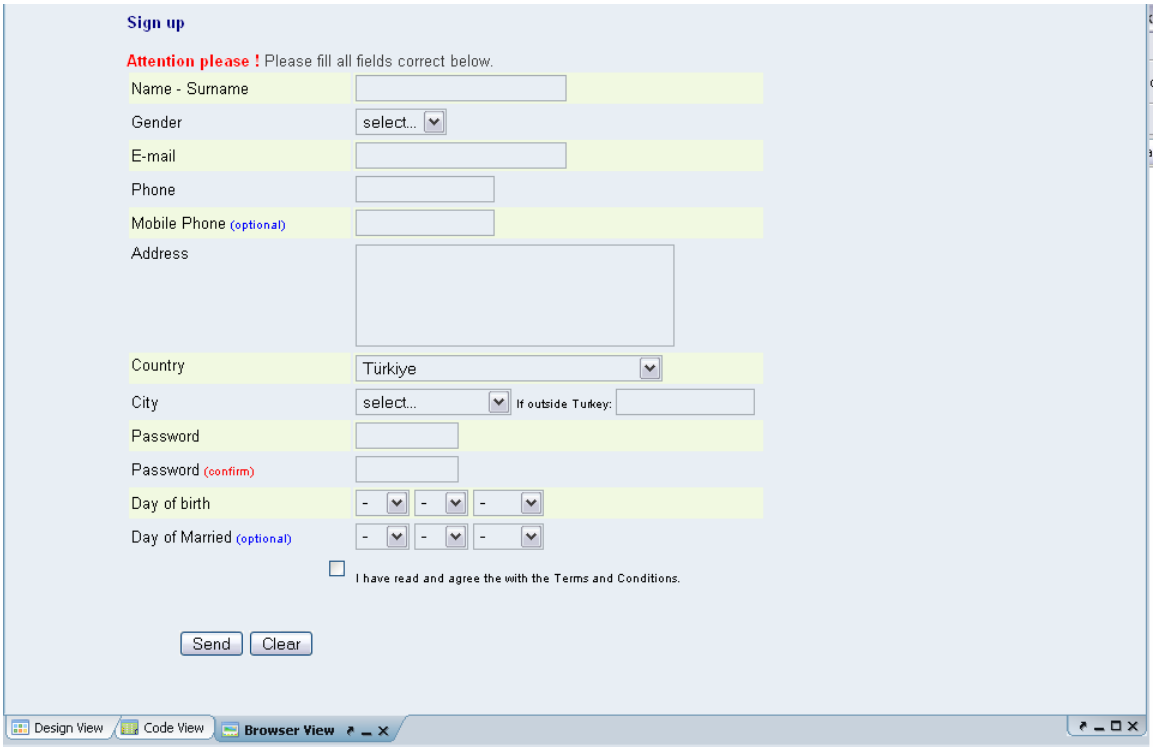


Figure 26 - Browser View

6.9. System Output

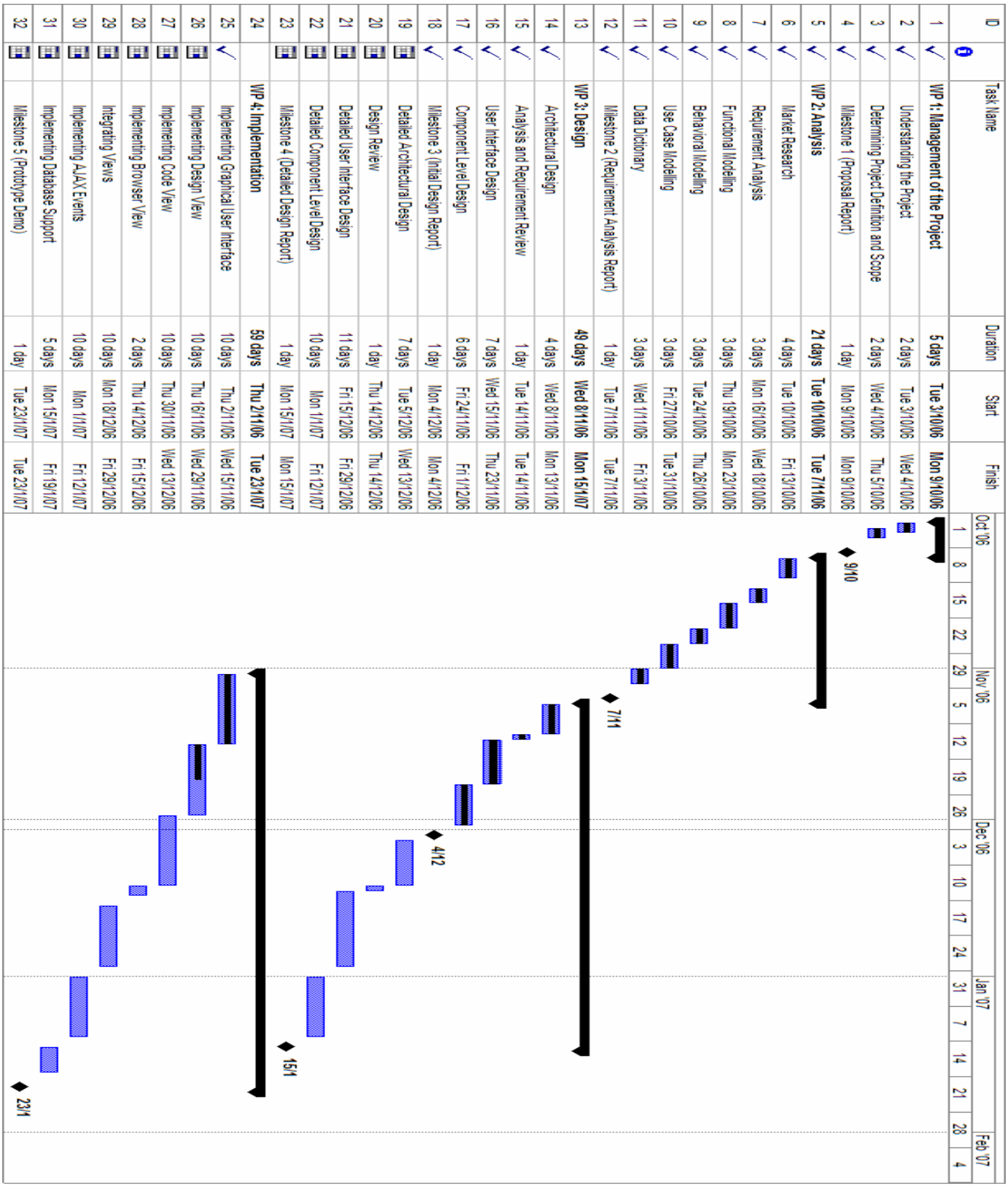
Normally the system output is minimized in the right bottom corner of the LADES. When the user do an action which results in standard output, i.e. runs the projects and returns error, it pops up from its place to the bottom of the LADES to display outputs.

7. Multilingual Support

In our project, as a team we always believe the importance of flexibility and user friendliness. So in LADES we will provide multilingual support. Therefore, the users can select the preferred language from the view part of the menu bar. In the view part, available languages will be displayed. The default language of the LADES program is English, however when the user chooses another language that language will be selected as default and the design will be shown in that language.

To provide that facility we will keep different files for each language in xml format. So when the user runs the program the related XML file will be parsed, and all the texts in the LADES program will be shown in that selected language.

8. Project Schedule



ID	Task Name	Duration	Start	Finish
1	**WP 1: Management of the Project**	**5 days**	**Tue 3/10/06**	**Mon 9/10/06**
2	Understanding the Project	2 days	Tue 3/10/06	Wed 4/10/06
3	Determining Project Definition and Scope	2 days	Wed 4/10/06	Thu 5/10/06
4	Milestone 1 (Proposal Report)	1 day	Mon 9/10/06	Mon 9/10/06
5	**WP 2: Analysis**	**21 days**	**Tue 10/10/06**	**Tue 7/11/06**
6	Market Research	4 days	Tue 10/10/06	Fri 13/10/06
7	Requirement Analysis	3 days	Mon 16/10/06	Wed 18/10/06
8	Functional Modelling	3 days	Thu 19/10/06	Mon 23/10/06
9	Behavioral Modelling	3 days	Tue 24/10/06	Thu 26/10/06
10	Use Case Modelling	3 days	Fri 27/10/06	Tue 31/10/06
11	Data Dictionary	3 days	Wed 1/11/06	Fri 3/11/06
12	Milestone 2 (Requirement Analysis Report)	1 day	Tue 7/11/06	Tue 7/11/06
13	**WP 3: Design**	**49 days**	**Wed 8/11/06**	**Mon 15/1/07**
14	Architectural Design	4 days	Wed 8/11/06	Mon 13/11/06
15	Analysis and Requirement Review	1 day	Tue 14/11/06	Tue 14/11/06
16	User Interface Design	7 days	Wed 15/11/06	Thu 23/11/06
17	Component Level Design	6 days	Fri 24/11/06	Fri 1/12/06
18	Milestone 3 (Initial Design Report)	1 day	Mon 4/12/06	Mon 4/12/06
19	Detailed Architectural Design	7 days	Tue 5/12/06	Wed 13/12/06
20	Design Review	1 day	Thu 14/12/06	Thu 14/12/06
21	Detailed User Interface Design	11 days	Fri 15/12/06	Fri 29/12/06
22	Detailed Component Level Design	10 days	Mon 1/1/07	Fri 12/1/07
23	Milestone 4 (Detailed Design Report)	1 day	Mon 15/1/07	Mon 15/1/07
24	**WP 4: Implementation**	**59 days**	**Thu 2/11/06**	**Tue 23/1/07**
25	Implementing Graphical User Interface	10 days	Thu 2/11/06	Wed 15/11/06
26	Implementing Design View	10 days	Thu 16/11/06	Wed 29/11/06
27	Implementing Code View	10 days	Thu 30/11/06	Wed 13/12/06
28	Implementing Browser View	2 days	Thu 14/12/06	Fri 15/12/06
29	Integrating Views	10 days	Mon 18/12/06	Fri 29/12/06
30	Implementing AJAX Events	10 days	Mon 1/1/07	Fri 12/1/07
31	Implementing Database Support	5 days	Mon 15/1/07	Fri 19/1/07
32	Milestone 5 (Prototype Demo)	1 day	Tue 23/1/07	Tue 23/1/07

9. Conclusion

During the preparation of the Initial Design Report, we draw diagrams such as class diagrams, sequence diagrams, activity diagrams, and use case diagrams. We can say that those diagrams are very beneficial for us to clarify many points about our project. In addition to this, in our project we add some new features such as “Multilanguage Support”, “Flexible Layout Design”, “Toolbar Customization”, and “Adding New (user defined) Web Application Components”.

In conclusion, as being ACME project team we believe the importance of the initial design report and we think that our initial design will be helpful to us and will simplify our design report and implementation phases.