International Journal of Computer Networks and Communications Security

VOL. 2, NO. 6, JUNE 2014, 202–210
Available online at: www.ijcncs.org
ISSN 2308-9830

Using the Component-Based Search Pattern to Enhance Reusing Product Discovery Mechanism in Mobile Adaptive E-Commerce WebApps

PARISA YOUSEFZADEHFARD[1], MINA ZOLFY LIGHVAN[2], MANOUCHEHR ZADAHMAD[3]

[1, 2] Department of Electrical and Computer Engineering, University of Tabriz, Tabriz, Iran

[3] Department of Computer Engineering, Ilkhchi Branch, Islamic Azad University, Ilkhchi, Iran

E-mail: [1]p.yousefzadeh91@ms.tabrizu.ac.ir, [2]mzolfy@tabrizu.ac.ir, [3]Zadahmad@iauil.ac.ir

ABSTRACT

E-commerce webapps provide people with comfortable ways to expose and order products. Ability to ubiquitously performing e-commerce activities like searching functionality is essential to e-commerce scenarios. The advent of smartphones is provided a novel insight into e-business. Unfortunately the innate features of handheld devices like different width and height size, memory capacity, and other software and hardware features lead to confuse customers on searching products or services and seeing results in a proper manner. To help better discovery of products using smartphones, this study provides best design practices (patterns) to supply searching abilities. The paper consists all component-based search patterns needed to discover the products and to show the results. For each search pattern; problem, motivation, context, forces and a solution is provided regarding capabilities of different smartphones. Finally the study discusses how using the collection of search patterns provide enhanced discovery mechanism in the context of smartphones via presenting known examples from well-known WeabApps.

Keywords: *E-Commerce, Smartphones, Search Patterns, Modeling, WebApps.*

1 INTRODUCTION

Software engineering patterns document proven solutions to repetitive software development problems in all activities of software development process. As it contains best practices used in significant projects, reusing these examined ideas help developers to increase the quality of resultant product and speed up the software development process. Distinguished names like Kent Beck and Ward Cunningham adapt this architectural concept (civil engineering) introduced by Christopher Alexander into software engineering that is culminated in the books‘ Design Patterns: Elements of Reusable Object-Oriented Software’ by Erich Gamma et al.[1], Pattern-Oriented Software Architecture (POSA) book series by Frank Buschmann et al [2], Patterns of Enterprise Application Architecture by Martin Fowler [3] and etc. Pattern Languages of Programs (PloP) is a group of annual conferences that is held in different continents to mature and develop the documenting and using patterns [4].

Both patterns and components try to reuse. They differ in the type of reuse. Components are stored in the form of source or binary codes but patterns are abstract models (not dedicated to a specific application). Some researchers try to store patterns in the form of components [5], despite this fact that the software engineering patterns need to be used in analyses and design actions. Some kind of patterns devoted to particular domains like user interface design patterns, web design patterns, and other domain-specific design patterns. Such patterns are likely to provide full or partial componentization [6, 7, and 8].

Search patterns are aimed to ease the fining of most relevant matches plus usability. Search pattern languages have extracted the most privilege instances of search functionalities from well-known e-commerce WebApps. In all of these documentations the user interface design is focused and is tried to be elaborated [9, 10, and 11]. This

study tries to provide design models to main search patterns which can be reference to develop search components especially for smartphones because of privilege smartphones in e-commerce scenarios.

For all e-commerce websites having a powerful searching ability is crucial. It is important to make sure of acceptable and usable presentation of search mechanisms in the handheld devices. This means software engineer must be aware of suitability of interaction mechanisms. The developer must consider helping customers to discover what they searching and showing result in appropriate format. For a website developer it is important to use best practices from the well-known web applications to achieve perfect user experience. This study discusses pattern group required to handle designing effective search including Explicit Search, Auto-Complete, Dynamic Search, Scoped Search, Saved & Recent, Search Form and Search Results. These Use Case diagram of search patterns presented in Fig. 1 The description of these patterns is presented in following sections.

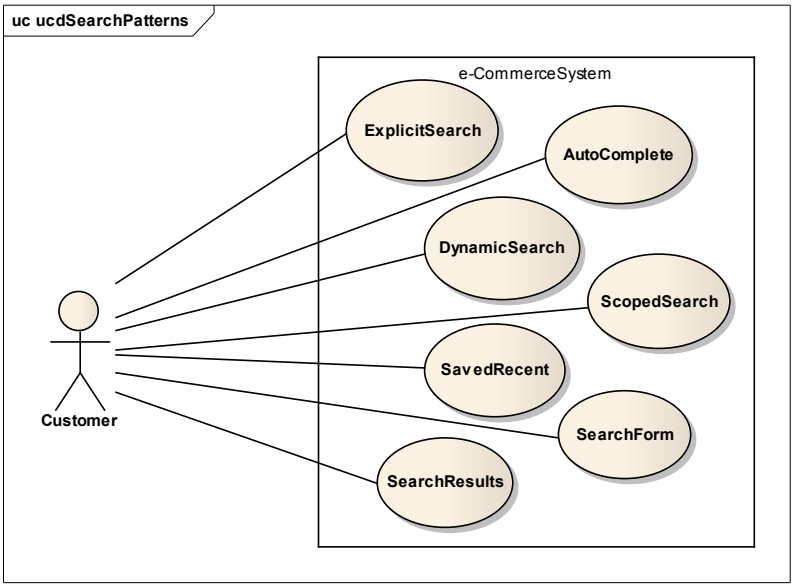


Fig. 1. Use Case diagram of search patterns

The rest of the paper as follow: Sections 2 through 7 documents the search patterns including Explicit Search, Auto-Complete, Dynamic Search, Scoped Search, Saved & Recent, Search Form and Search Results. Each description for patterns encompasses Context on which pattern emerged, Motivation for use the pattern, Problem to be solved, Forces to be considered, solution and the classes and their collaboration in the form of UML sequence and class diagrams. Finally section 8 discusses the effect of collection of search patterns in developing e-commerce systems.

2 EXPLICIT SEARCH

Context: Rather than using navigation to acquire information or content, the user wants to do a direct search through content contained on multiple Web pages.

Motivation: Lack of a powerful searching ability or existence of a complex search page wind up customers who are less likely to spend the time in your website.

Problem: The users require to search a product or specific information within a web site. Having an explicit search feature on the website is essential to satisfy this customer's need.

Forces: Users may want to search for an item in a category. Additionally user might prefer to further specify a query.

Solution: Create a search module in the website. It is accomplished by using a search label, a search space for typing words and a mechanism for starting the search action that leads to design that is called Explicit Search Pattern. Fig. 2 present the examples from known websites for Explicit Search.

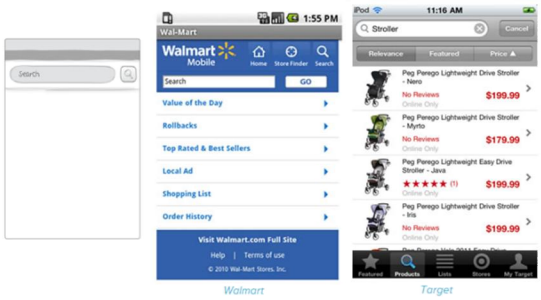


Fig. 2. Explicit Search and its known examples

Participants and Dynamics: As illustrated in fig. 3, Customer's click on Go button or Enter Key rises an event that is handled by btn_Go_Click event handler of SearchUI boundary (UI) object. Subsequently SearchUI calls the doExplicitSearch(string) method of SearchBLogic control (Business Logic) object and passes the search query to it. Then the algorithm within the method is executed and as a result the search results are returned in the form of HTML_SearchResult object.

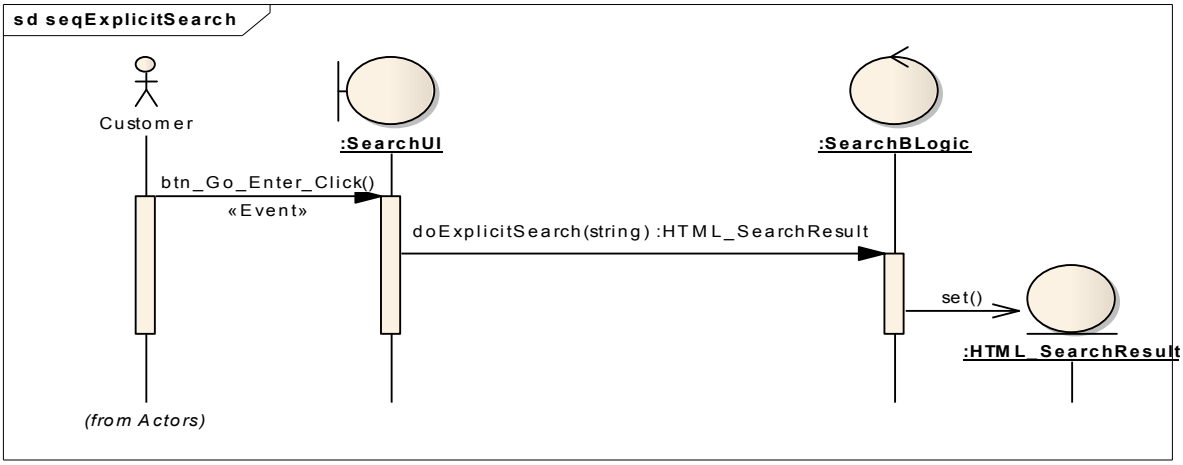


Fig. 3. Participants and Dynamics in Explicit Search pattern

3 AUTO-COMPLETE

Context: Naturally, auto-complete is always part of a Form in websites especially search forms in an e-commerce WeabApps.

Motivation: The user may want to quickly select the desired search value within the set of possible and potential values.

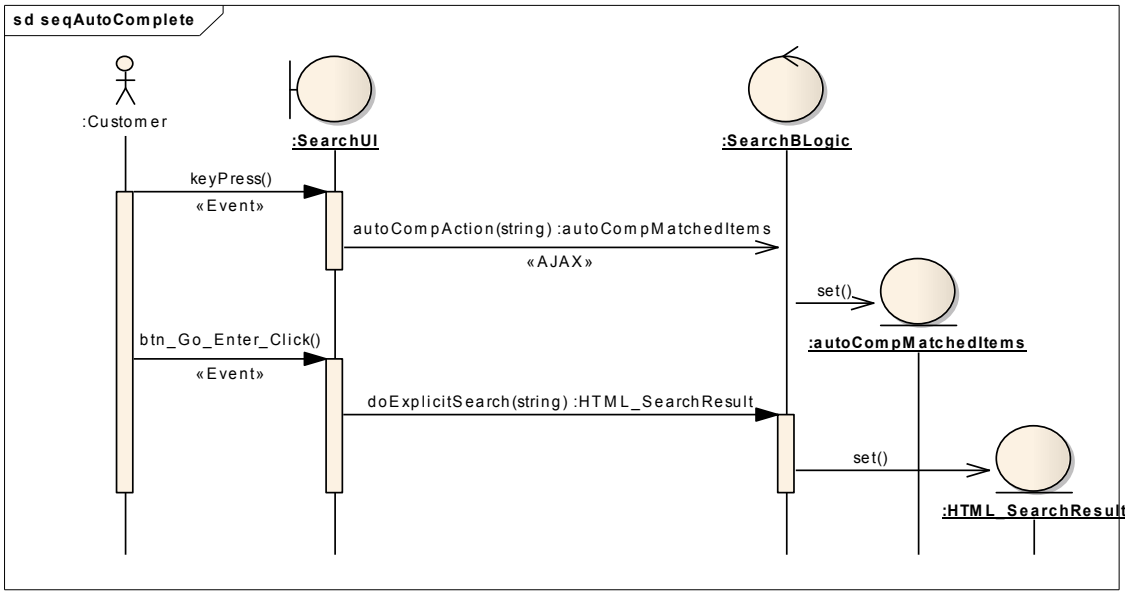
Problem: The miss-type is one of the challenges that users in WebApp encounter with. In the other hands some products or search phrases are hard to remember.

Forces: providing continues feedback loop to user to narrow in on the correct choice require considering different design mechanisms in different devices.

Solution: With every user key press, the application starts comparing with the search set and create a match list that is shown underneath the search box. It makes possible to user to select one of the items. Fig. 4 present examples from known websites for Auto-Complete.

Fig. 4. Auto-Complete and its known examples

Participants and Dynamics: As illustrated in fig. 5, any Customer's key press rises an event that is handled by client side keyPress event handler of SearchUI boundary (UI) object. Subsequently SearchUIuses the AJAX technology to call the autoCompAction(string) method of SeachBLogic control (Business Logic) object and passes the search query to it. As a result of this call, matched items returned in asynchronous manner and appeared in the search box. By Customer's click on Go button or pressing Enter key the algorithm within the method is executed and as a result the search results are returned in the form of HTML_SearchResult object.



4 DYNAMIC SEARCH

Context: In a fast internet connections users wants to immediately see the search results as they enter every letter.

Motivation: Users prefer to do not type whole search query. As well as users may not remember the complete search query.

Problem: Searching through an extensive list of items and on a long dataset when the user does not remember the search term is a big challenge in WebApps.

Forces: Providing dynamic filtering need considering abilities of different devices. Not all device are that much fast to update the screen with new results. The search provider must create device adaptive rendering considering the capabilities of different devices.

Solution: Provide a search form that dynamically filters the results as the user is typing. Fig. 6 present examples from known websites for Dynamic Search.

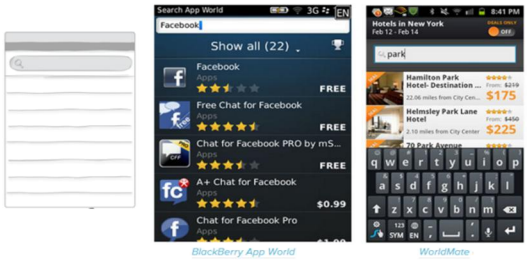
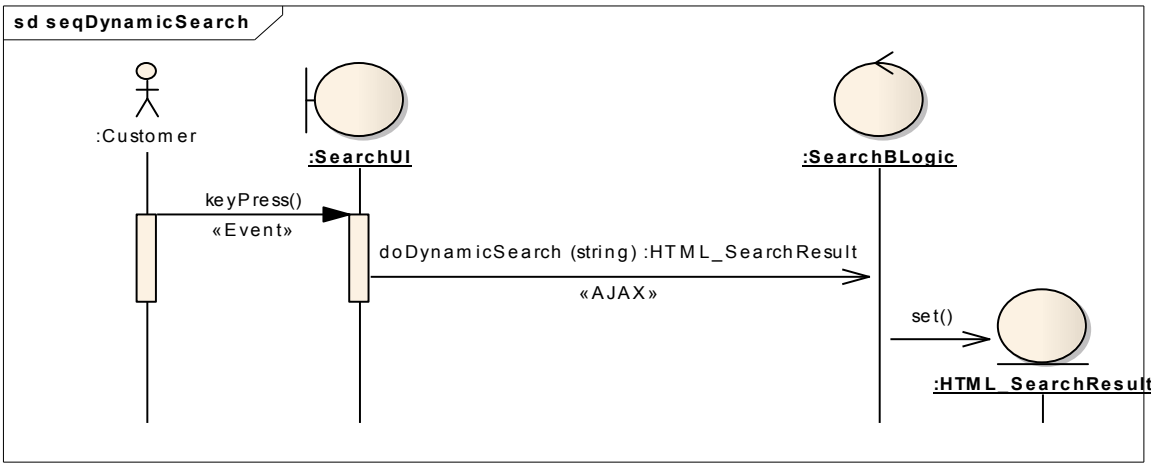


Fig. 6. Dynamic Search and its known examples

Participants and Dynamics: As illustrated in fig. 7, any Customer's key press rises an event that is handled by client side keyPress event handler of SearchUI boundary (UI) object. Subsequently SearchUI uses the AJAX technology to call the doDynamicSearch (string) method of SeachBLogic control (Business Logic) object and passes the search query to it. As a result of this call, search results are returned in the form of HTML_SearchResult object in asynchronous manner and appeared in the search box. This pattern do not need to Go button or Enter key events, though these events can be used in addition it.



5 SCOPED SEARCH

Context: In an e-commerce WebApp there are predefined categories of items. Sometimes user is not interested to all of these categories.

Motivation: Categorizing the products helps the customer to focus on a special category.

Problem: How to provide an easy and fast mechanism to get to the desired result before performing the search?

Forces: It is not all time possible to have an appropriate categorization list that satisfy all users. The Scope-list must be available to user in different devices according to its capabilities.

Solution: Show the Scope-list in an appropriate location near the search box. It is also helpful to provide suitable icon to every scope item. Fig. 8 present examples from known websites for Scoped Search.

Fig. 8. Scoped Search and its known examples

Participants and Dynamics: The dynamics are similar to Explicit Search pattern with a simple difference that selecting any scope or categories in addition to filter the search results also rises the search event.

6 SAVED & RECENT SEARCHES

Context: In an e-commerce WebApp which provides search functionality to enormous number of users the response time is a crucial quality attribute. There is also an immense number of daily similar searches that is queried by same or different users.

Motivation: Saving the privilege recent search queries and search results in the memory can please both the customer and e-commerce companies. Customers will receive the results very fast without need to reenter the search query. E-commerce companies can support more customers using this approach.

Problem: How to provide appropriate practices to save and restore search queries and results to respect user's effort?

Forces: The way of selecting from previous searches in different devices may be the issue of device adaptiveness. The e-commerce system need to render the selecting ability in a device adaptive manner.

Solution: In addition to available search practices, the Saved Search require methods to store searches in database and restore it later. In the cases where search results may be different for a similar query in different times the pattern need to update the results in a specified times. Fig. 9 present examples from known websites for Saved & Recent Searches.

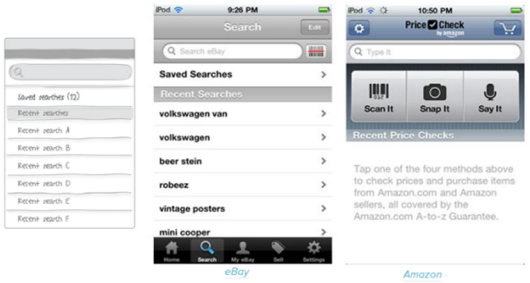


Fig. 9. Saved & Recent Searches and its known examples

Participants and Dynamics: As illustrated in fig. 10, Customer's click on Saved Searches link rises an event that is handled by linkSavedSearch() event handler of SearchUI boundary (UI) object. Subsequently SearchUI calls the showSavedSearches() method of SeachBLogic control (Business Logic) object. Then the saved search results are returned in the form of HTML_SearchResult object.

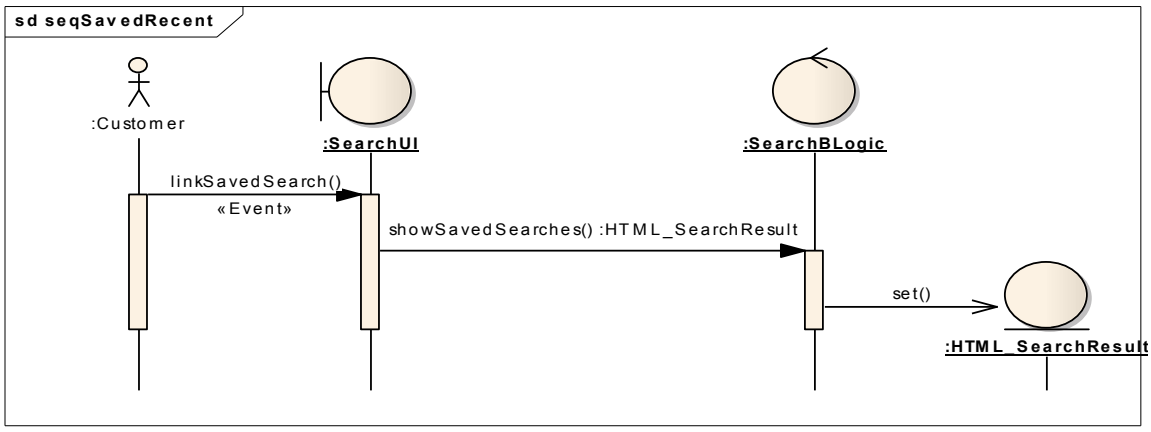


Fig. 10. Participants and Dynamics in Saved & Recent Searches pattern

7 SEARCH FORM (ADVANCED SEARCH)

Context: In an e-commerce system which provide search functionality, the customer may want to elaborate the search query by specifying some search criteria.

Motivation: Proving an advanced search that in a user friendly manner in a single page like Explicit Search Pattern is more interesting to user.

Problem: How to make it possible to user to narrow down the search results by embedding different criteria in search query.

Forces: To achieve better user experience only the essential fields must be available to specify the search criteria.

Solution: This pattern uses a form to select all search criteria. Following best design practices (alignment, labels, and size) leads to better user experience. Fig. 11 present examples from known websites for Search Form.

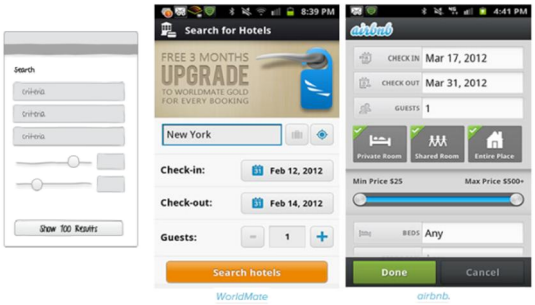


Fig. 11. Search Form and its known examples

Participants and Dynamics: As illustrated in fig. 12, Customer's click on Go button or Enter Key rises an event that is handled by btn_Go_Enter_Click(string) event handler of SearchUI boundary (UI) object. The parameters set by Customer in search form is collected in client-side and is sent to the server-side as argument of btn_Go_Enter_Clickevent handler. Subsequently

SearchUI calls the doAdvancedSearch(string) method of SeachBLogic control (Business Logic) object and passes the search query together with search parameters to it. Then the algorithm within the method is executed and as a result the search results are returned in the form of HTML_SearchResult object.

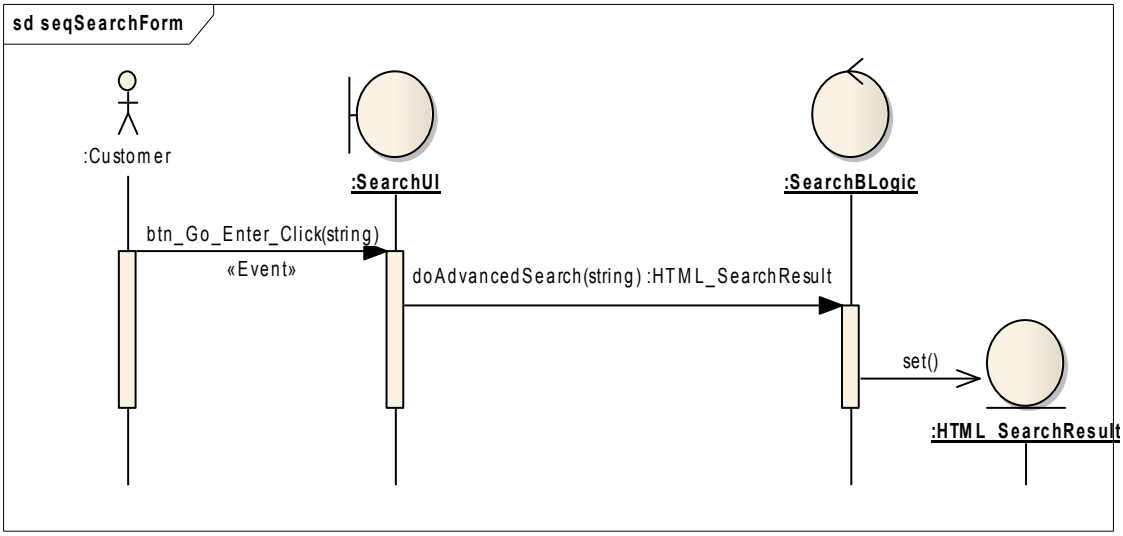


Fig. 12. Participants and Dynamics in Search Form pattern

8 SEARCH RESULTS

Context: In an e-commerce WebApp with a search functionality, it is needed to show results in an appropriate format.

Motivation: User need to investigate the result by clicking on search results to see the detail of items and maybe add it to a Shopping Cart.

Problem: How to provide user methods to process a list of search results?

Forces: It is required to provide easy ways to users to scan search as well as to reformat the results according to their preference.

Solution: A numbered list of results starting with the most relevant result provided with a description are returned to the user (fig. 13). Most often results are presented in the form of a fixed number of results per page plus mechanisms for Paging. The format of page showing the search results may differ regarding the results type (image, video, descriptive or etc.).

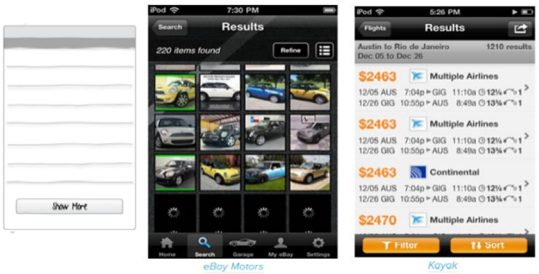


Fig. 13. Search Results and its known examples

Participants and Dynamics: Fig. 14 shows the structure of the Search Result structure and its relationship with other participants. The Search Results class represents the abstraction of the results produced as a result of search module. Results can be of descriptive, image or video types. In all result types a series of contents which are stored in the form of Composite Pattern [@Gamma] are rendered and integrated together to constitute the final Search Results. Each SearchResult objectdeclares the interfaces for

objects in the search result composition, an interface for accessing and managing its child components, an interface for accessing a component's parent in the recursive structure, and implements default behavior for the interface common to all classes. The fig. 14 also illustrates the relationship among SearchUI, SearchBLogic, and SearchResults classes. The SearchBLogic uses SearchResults as a client and calls deviceAdaptiveResults to adapt final results according to characteristics of connected mobile device.

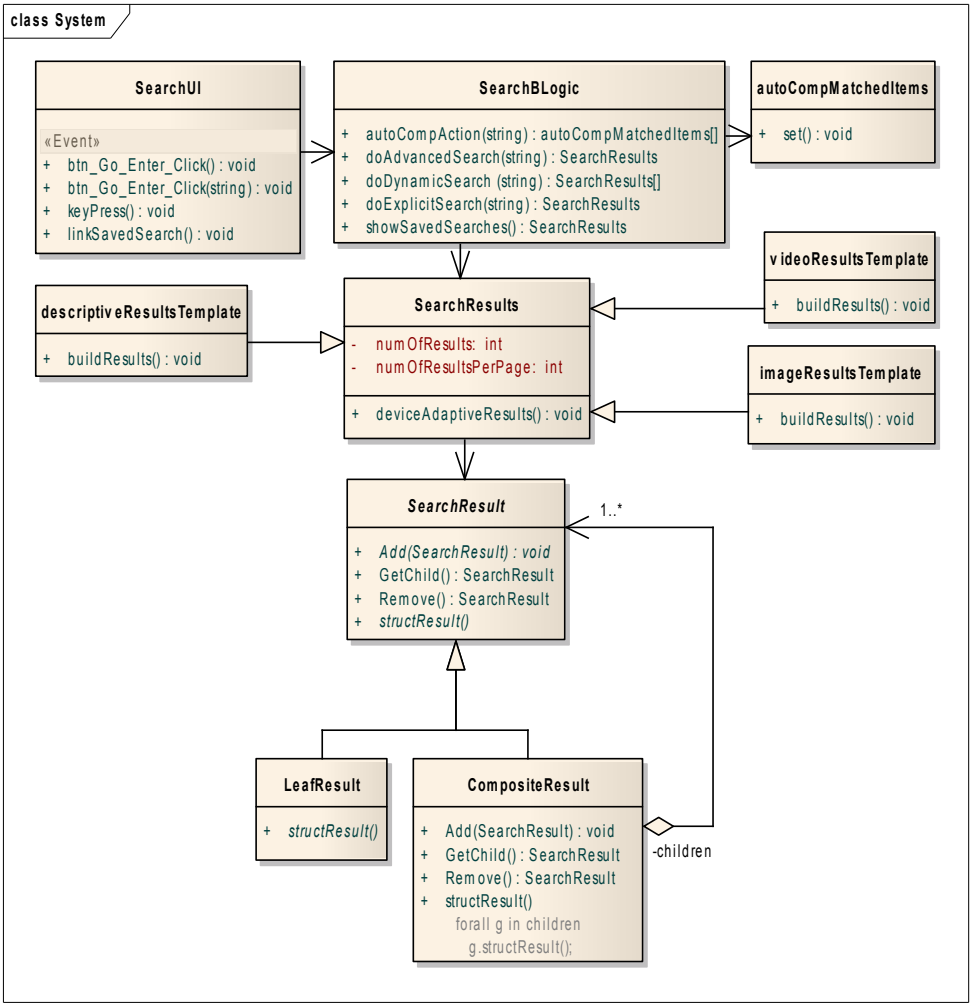


Fig. 14. Participants and Dynamics in Search Results pattern

9 DISCUSSION

This work tries to make a look on search patterns from the component based development viewpoint. It also considered the application of search patterns in the context of e-commerce WebApps especially through smartphones. Not only a complete documentation is provided to major search patterns, but also a design model including the structural and dynamic aspect also provided.

The relation among the real-world example, theoretical reasoning and modeling is correlated. Such relation help user to understand how transfer search ideas to implementations. For example auto-complete is one of common search patterns but just technology specialists are aware of its need to AJAX technology. The presented sequence diagram completely make it clear when and how to use this technology to achieve the desired outcomes.

To adapt the content to different characteristics of handheld devices an adoption functionality is suggested. It fetch the attribute of client devices and render the UI of search patterns according to this information. For example the display width and height, graphical capabilities, CPU speed, memory capacity, browser features, plus sensing data is

crucial in producing an adaptive search UIs. These altogether lead to better design quality and performance.

Simple search results are in a template like formats. For example descriptive search results contains a title for a search results that at same time is a hyperlink to a document or a part of a document plus two or three most relative description the match to the search query. But real e-commerce systems prefer to illustrate the results in more attractive contents. Such contents comprise some graphical primitives like texts and some composites objects which in turn recursively contains primitives and composites. This study managed this structural complexity using Composite Pattern [1]. This pattern helps device adaptive functionality to treat primitive and composite objects uniformly.

10 REFERENCES

[1] Erich Gamma, Richard Helm, Ralph Johnson, and John Vlissides. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, 1995, ISBN 0-201-63361-2.

[2] Frank Buschmann, Regine Meunier, Hans Rohnert, Peter Sommerlad. Pattern-Oriented Software Architecture, Volume 1: A System of Patterns. John Wiley & Sons, 1996, ISBN 0-471-95869-7.

[3] Martin Fowler. Patterns of Enterprise Application Architecture. Addison-Wesley, 2002, ISBN 978-0-321-12742-6.

[4] The Hillside Group – Pattern Languages of Programs (PloP) conference series: http://www.hillside.net/plop.

[5] Bertrand Meyer, Karine Arnout . "Componentization: The Visitor Example". IEEE Computer (IEEE) (39/7), July 2006, 23–30, doi:10.1109/MC.2006.227.

[6] Manouchehr ZadahmadJafarlou, Ali Moeini, Parisa YousefzadehFard, New process: pattern-based model driven architecture, Procedia Technology (1), 2012, 426–433, DOI: 10.1016/j.protcy.2012.02.095.

[7] Manouchehr ZadahmadJafarlou, Bahman Arasteh, Parisa YousefzadehFard, OO Divide and Conquer Pattern Suitable for Parallel, Grid and Cloud Computing, Lecture Notes in Electrical Engineering (114), 2012, 487-495,

[8] DOI: 10.1007/978-94-007-2792-2_46.

[9] Manouchehr Zadahmad Jafarlou, Parisa Yousefzadeh Fard, Heuristic and pattern based Merge Sort, Procedia Computer Science, Volume 3, 2011, Pages 322–324. DOI: 10.1016/j.procs.2010.12.055.

[10] [9] Douglas K. van Duyne, James A. Landay, Jason I. Hong, Design of Sites: Patterns, Principles, and Processes for Crafting a Customer-Centered Web Experience, Addison Wesley, 2002, ISBN : 0-201-72149-X.

[11] Greg Nudelman, Designing Search: UX Strategies for ECommerce Success, Wiley Publishing, Inc. 2011, ISBN: 978-0-470-94223-9.

[12] Peter Morville, Jeffery Callender, Search Patterns, O'Reilly Media, Inc. 2010, ISBN: 978-0-596-80227-1.