Proc. of the Second Intl. Conf. on Advances In Computing, Communication and Information Technology- CCIT 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-051-4 doi: 10.15224/ 978-1-63248-051-4-137

Menu-based Service Access and Delivery Pattern

Towards Achieving Equitable Access to Digital Services

Ishmael Makitla

***Abstract*— The proliferation of mobile phones in developing countries such as South Africa has been revolutionary. This might have sufficed to address the issues of digital divide but not the digital difference. The latter means that not all these mobile phones have equal technological capabilities and this in turn has impact on the nature of the functionality that the mobile phone affords to its user. The basic capability of a mobile phone is the ability to process and display textual information. This capability is used in basic service such as Short Message Service (SMS) as well as Unstructured Supplementary Service Data (USSD). There is a need for a service access and delivery pattern that can facilitate the delivery of services using common-lowest denominator in terms of technological capabilities of mobile phones. This paper describes the conceptualization and reference implementation of a menu-based service access and delivery pattern. The reference implementation has demonstrated that through the menu-based pattern, the same content traditionally accessible only to Smartphones can be made available to low-end mobile phones as well.**

***Keywords*—mobile phone, menu, USSD, service delivery,pattern**

I. Introduction

Access to technology is a strong driver for the development of the information society [1]. In many developing countries, including South Africa, access to ICTs by the general population is poor. A large portion of the population have no access to computers – this results in what is known as the digital divide, where certain groups have good access to technologies, whereas other groups (typically the poor) do not. The digital divide is a strong inhibitor to the development of the information society.

The proliferation of mobile phones in South Africa (and other African countries) has been remarkable – there are now more Africans with access to mobile phones than before. It is estimated that more than 40 million people in South Africa alone own or have access to a mobile phone. This presents an opportunity to address the problem of the digital divide. However even with these positive developments in the proliferation of mobile phones equitable access to Information and Communication Technology (ICT) services is still hampered by socio-economic challenges such as affordability of technologically advanced ICT devices such as SmartPhones. This results in digital difference which has the impact of limiting access to equality of opportunities through ICTs, specifically value-added services [2].

Ishmael Makitla

Council for Scientific and Industrial Research (CSIR)
South Africa

The key challenge when delivering value-added services to resource-constrained end users is dealing with differences in technological capabilities (digital difference) of end user communication devices such as mobile phones[3]. In other words, a mechanism must be found or developed to enable service providers to deliver value-added services to end-users regardless of the access technologies or devices used to access these services. The said services should be built once and immediately made available through multitude of access channels without the service providers having to develop a new instance of such a service for each available access channel. An approach to service delivery design that requires dedicated infrastructure for each market segment or technology-type user group is called stove pipe architecture and is very costly [4]. In responding to the need to achieve equitable access to mobile digital content and service, an access-channel agnostic service access and delivery mechanism has been conceptualized and implemented.

Menu-based services have traditionally been associated with mobile Unstructured Supplementary Service Data (USSD) such as those used for recharging air-time or checking balances as well as mobile banking. The access-technology agnostic design pattern means that the same menu-based service experience can be delivered through different channels, including Interactive Voice Response (IVR) systems, Instant Message chat platforms (Mxit, Gtalk, etc.). This is made possible by the design which was followed in the development of the delivery platform. Furthermore, access-technology agnostic design pattern implementation addresses the digital difference and thus has the potential to extend the reach and impact of digital services and content. The technology is described in more details in Section II.

The remainder of this paper is organized as follows: **Section II** discusses and demonstrates the technical viability of the conceptual principles that underpin the menu-based service access and delivery pattern. **Section III** discusses the Menu Action Specification Framework which is used in menu-based service access and delivery to encapsulate the computation logic. Section **IV** discusses how this research work is different from existing menu-based related research work. The paper is concluded in **Section V**.

II. Menu

To guide the discussion on the menu-based service access and delivery pattern, three main components are identified:

- The *Conceptual Model* which captures the domain concepts - what is a menu, what are possible menu options, what types of menu options should there be, and how do these menus and menu options relate to business functionality.

Proc. of the Second Intl. Conf. on Advances In Computing, Communication and Information Technology- CCIT 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-051-4 doi: 10.15224/ 978-1-63248-051-4-137

- The *Menu-based Service Access and Delivery Design Pattern* which describes how a menu-based service access and delivery system should be developed. It captures the main components of such a system in an architectural format – what components are necessary and how they interact to deliver the menu-based service.
- *The technology-specific reference implementation* which illustrates the technical viability of the design pattern and the architecture as well as the ability to use existing standards (JSON or XML) to implement the conceptual model.

A. *Conceptual Model*

Conceptually, a Menu is discernible through the use of a tree-analogy. Accordingly a Menu has a Root, Branch(es) and Leaf(s). The Root of a Menu is the top-most element that is not attached to any other menu but has other menus attached to it instead. A Branch on the other hand is the type of a Menu that is attached to another Menu and has other menus attached to it. A Leaf is the termination point in the tree analogy and marks the end of the navigation path. A Leaf is attached to a Branch and has no other part of the tree attached to it.

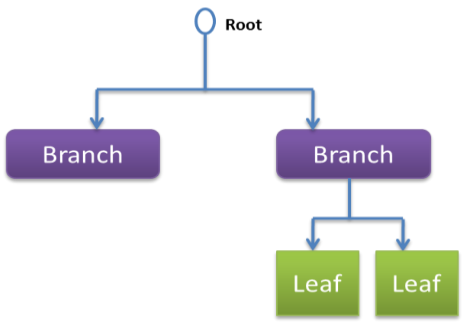
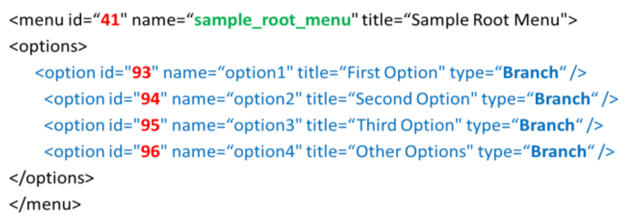


Figure 2 Menu tree analogy

A sample menu definition is depicted in Figure 3 and shows a menu with four options of type Branch.



```
<menu id="41" name="sample_root_menu" title="Sample Root Menu">
<options>
    <option id="93" name="option1" title="First Option" type="Branch" />
      <option id="94" name="option2" title="Second Option" type="Branch" />
      <option id="95" name="option3" title="Third Option" type="Branch" />
      <option id="96" name="option4" title="Other Options" type="Branch" />
</options>
</menu>
```

Figure 3 Sampel Menu definition

In a Menu-based system the application logic should be encapsulated in the Menu of type Leaf. Essentially, the Leaf should embed the piece of logic that needs to be executed when the user selects that particular Leaf from the list of options. Because of its centrality to this work, Menu Action Specification Framework is discussed in more detail in Section III.

B. *Menu-based Service Access and Delivery Design Pattern*

The design pattern is depicted in Figure 1 and clearly indicates how the Conceptual Model is being used in the overall system architecture

The key components of the menu-based service access and delivery design pattern are described below:

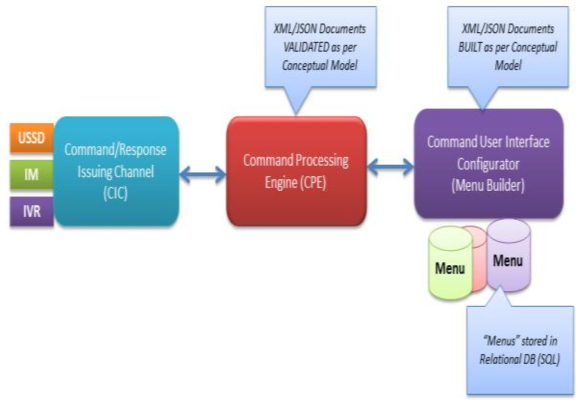

XML/JSON Documents VALIDATED as per Conceptual Model
XML/JSON Documents BUILT as per Conceptual Model
USSD
IM
IVR
Command/Response Issuing Channel (CIC)
Command Processing Engine (CPE)
Command User Interface Configurator (Menu Builder)
Menu
Menu
"Menus" stored in Relational DB (SQL)

Figure 1. Service Access and Delivery Design Pattern

- *Command Issuing Channels* (access/delivery channels) abstract complexities of the underlying access networks through which commands (menu-based service requests) are issued and where responses are rendered back to the users (e.g. USSD, Instant Messaging or IVR)
- *Command Processing Engine* (menu processor) is programming language independent (XML/JSON) – it validates in-coming menu documents in accordance with the Conceptual Model for Menus.
- *Command User Interface Configurator* (menu builder) is also language independent – any technology can be used to build Menus according the Conceptual Model, these Menus are then stored in database storage of choice.

C. *Technology-specific reference implementation*

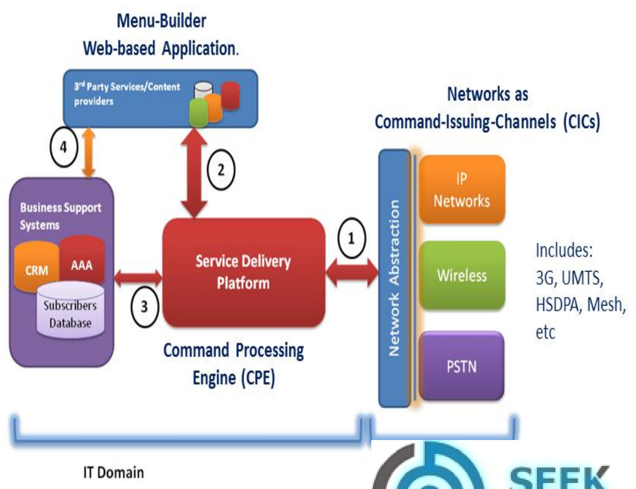


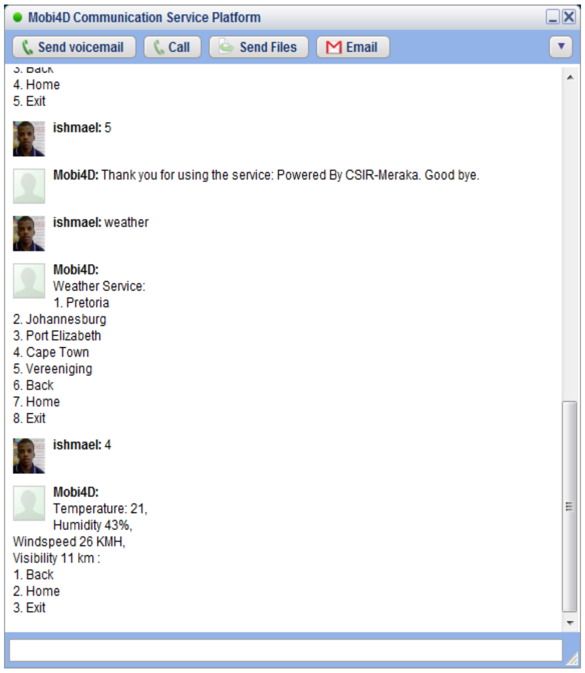
Figure 4 Menu-based Service Access and Devlie

Proc. of the Second Intl. Conf. on Advances In Computing, Communication and Information Technology- CCIT 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-051-4 doi: 10.15224/ 978-1-63248-051-4-137

In terms of the technology-specific instance of the menu-based service access and delivery design pattern, Makitla and Fogwill in [3] illustrate the implementation of menu-based service access and delivery using the telecommunications' Service Delivery Platform (SDP) [5] concept. Figure 4 illustrates how the components of the design pattern can be mapped to components of the SDP:

When the user sends a command through the connectivity infrastructure (the networks), this commands is received through the Command-Issuing-Channel (CIC) and passed to Command Processing Engine which will query the Menu storage for the menu that corresponds to the issued command. The Menu, depending on its type (Root, Branch, Leaf) is processed and appropriate feedback is sent to the user –this might be a menu with a list of options.

Figure 5 shows a typical user interaction with using GoogleTalk (GTalk) chat client as a CIC to access menu-based weather service.



```
5. Back
4. Home
5. Exit
ishmael: 5
Mobi4D: Thank you for using the service: Powered By CSIR-Meraka. Good bye.
ishmael: weather
Mobi4D:
Weather Service:
1. Pretoria
2. Johannesburg
3. Port Elizabeth
4. Cape Town
5. Vereeniging
6. Back
7. Home
8. Exit
ishmael: 4
Mobi4D:
Temperature: 21,
Humidity 43%,
Windspeed 26 KMH,
Visibility 11 km :
1. Back
2. Home
3. Exit
```

Figure 5: Menu-based Weather Service access using GTalk.

The Menu-based weather service has been used as a demonstrator to show how a service that is traditionally accessible only through the web, and thus excluding users of SMS-Call-only mobile phones, can be made accessible through a menu-based service access and delivery platform.

The use case such as the one depicted in Figure 5 is made possible by the ability to define specific menu actions to be executed when a user selects an option on the menu. For this purpose, we have developed a Menu Action Specification Framework.

III. Menu Action Specification Framework

Menu Action Specification instructs the menu/command processing engine to take a specific action when the user selects the menu option of type Leaf on which this Menu Action Specification is attached. A menu action is identified by its Identifier (ID), its Name and its Type.

```
<action id="5" name="getDurbanWeatherviaREST">
        <action-type>REST</action-type>
        <action-operand>
                http://www.webservicex.net/globalweather.asmx/GetWeather?
        </action-operand>
        <action-data>
                CityName=Durban,CountryName=South Africa
        </action-data>
</action>
```

Figure 6: Menu Action Specification

Any menu action requires something with which to perform the task. For this, the Menu Action Specification defines an *operand* and *data* as its main elements. The value of the operand is determined by the action type which also informs the interpretation of the Data element. Figure 6 depicts a sample definition of a Menu Action Specification used to retrieve weather information. The action is of type Representation State Transfer (REST), and its operand is a Universal Resource Locator (URL) pointing to the weather service provider, and the data hold a set of parameters to be passed to the URL to specify the city and the country for which the user request weather information.

Because the action is executed on the server side, it is not limited to the capabilities of the end-user mobile phone. In this way, it is possible for a user on a low-end mobile phone to access the same weather content through USSD.

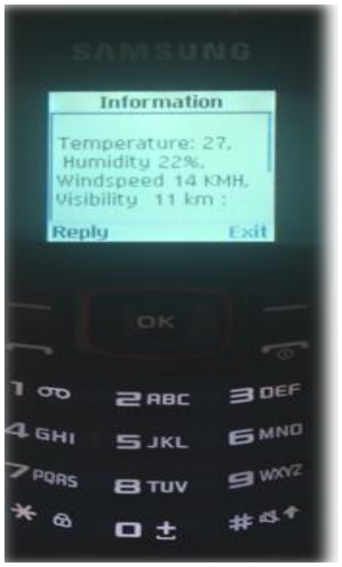


Figure 7 Weather service via USSD

Proc. of the Second Intl. Conf. on Advances In Computing, Communication and Information Technology- CCIT 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-051-4 doi: 10.15224/ 978-1-63248-051-4-137

Furthermore, the Menu Action Specification framework is completely descriptive – it describes what needs to be done (action) and not how this should be done. It is technology independent – the developer of a menu-based application can use whichever technology they prefer to implement the action types.

A full menu definition with a Menu Action Specification attached to one of its options is depicted in Figure 8:

```
<menu id="93" name="option1" title="First Option">
<options>
    <option id="10" name="option1Leaf" title="First Option Leaf" type="Leaf">
    <actions>
            <action id="5" name="option1ActionDisplayMessage">
            <action-type>TERMINAL</action-type>
            <action-operand>$USER</action-operand>
            <action-data>
            See Action-Types for the kind of data to put here...
            </action-data>
            </action>
   </actions>
  </option>
  <!-- Other Options go here -->
</options>
</menu>
```

Figure 8 Menu definition with Menu Action Specification

A. *Menu Action Variables*

In addition to the operand and data element of the menu action, we also defined standard variables which are resolved to actual values during a menu session. The $USER resolves to the identifier of the user who is current navigating the menu – this depends on the channel and can be a mobile phone numbers, and email or instant messaging identifier.

The $MENU refers to the current menu the user is viewing. $OPTION refers to an option on the menu. The variable $PROMPT is used to set a value for either an operand or data element of the action – it tells the menu processing engine that it needs to prompt the user for input. These variables are used in menu based applications that collect information from the user – such as conference feedback.

As an example, suppose you defined an action of type NOTIFIER whose operand is set to “$PROMPT person to notify” and the data is set to “$PROMPT notification to send”. When the user selects the option on which this action is attached, the user will be asked to set value for “person to notify” and “notification to send” – only after these values are provided will the menu processing engine execute the action.

B. *Menu addressing and navigation*

A menu is uniquely identifier by its identifier (ID) and its name. The name or ID of a menu is used to retrieve the menu. For USSD, it is easier to refer to menus by their IDs, but in other channels, like instant messaging, it makes sense to use easy-to-remember menu names. Because menus are unique, they can be addresses directly without the needs for the user to traverse their parent menus. This is important design decision because it makes our menus exactly one level deep (see Figure 3)

Using USSD, for instance the user dials *120*2747*<menu-id>#

The fact that menus can be addressed directly also means that they are self-contained and therefore can be reused. To reuse a menu in another menu – the designer simply adds a menu option with its ID referring to the menu he wants to reuse.

IV. Related Work

This paper was prompted in part by the lack of reference work in menu-based service access and delivery. During the exploratory stage of the research, we found very little evidence of works that focused on menu-based computing. Vumi [6] seems to offer most of the capabilities discussed here except for the sound conceptual model, the concept of Menu Action Specification and the support for direct menu addressing and reusability. Another closely related work is the patent application from Research in Motion (RIM) [7]– but this discusses the context menu design for mobile phones and not using menu as service access and delivery pattern.

V. Conclusion

This paper has introduced the concept of menu-based service access and delivery pattern. The service access and delivery pattern takes advantage of the lowest common denominator of mobile phone capabilities (text processing) and uses this to address technical issue of equitable access. The paper described the conceptual model that underpins the menu-driven computing; it also described the service access and delivery pattern in more detail, including illustrating how it can be mapped to the telecommunications’ Service Delivery Platform (SDP). Screenshots are depicted in the paper to show that the concepts discussed in this paper are technically viable and have been implemented, and are actually in use.

The lack of referenced material on menu-driven computing and menu-based service access and delivery more specifically motivated the submission of this paper. The paper contributes a missing piece in the understanding of menu-based applications and services by providing a good theoretical model. An important technical contribution with regards to menu-based computing is the introduction of the technology-independent Menu Action Specification Framework which describes how menus encapsulate the computational logic of an application.

References

[1] A. Botha, M. van der Berg, J. Batchelor, and C. Islas Sedano, “Ability through mobility,” in *the 3rd International Conference on e-Learning*, 2008, pp. 43–48.

[2] J. C. AKER and I. M. MBITI, “Mobile Phones and Economic Development in Africa,” *J. Econ. Perspect.*, vol. 24, no. 3, pp. 207–232, 2010.

[3] I. Makitla and T. Fogwill, “Mobi4D: Mobile value-adding service delivery platform,” in *Lecture Notes in Electrical Engineering*, 2012, vol. 107 LNEE, pp. 55–68.

Proc. of the Second Intl. Conf. on Advances In Computing, Communication and Information Technology- CCIT 2014.
Copyright © Institute of Research Engineers and Doctors, USA .All rights reserved.
ISBN: 978-1-63248-051-4 doi: 10.15224/ 978-1-63248-051-4-137

[4] S. MAGEDANZ, T., BLUM, N. DUTKOWSKI, "Evolution of SOA Concepts in Telecommunications," *Computer (Long. Beach. Calif).*, vol. 40, pp. 64–68, 2007.

[5] H. LU, Y. ZHENG, and Y. SUN, "The Next Generation SDP Architecture: based on SOA and integrated with IMS.," in *Second International Symposium on Intelligent Information Technology Application (IITA'08)*, 2008, pp. 141–145.

[6] "Vumi," 2014. [Online]. Available: http://vumi.org. [Accessed: 01-Oct-2014].

[7] T. Barnes and A. Nguyen, "Apparatus and Method For Presenting Menu Items on User Interface of Consumer Electronics Device," US 8,438,501 B2, 2013.