Towards an Effective Distribution of Subsidized Food Using Mobile-based e-Government Solution

Osman Ibrahim
Faculty of Informatics & Computer Science
British University in Egypt (BUE), El Shourouk City, Cairo, Egypt
Osman.ibrahim@bue.ed.eg

Abstract —**Traditional means for the distribution of subsidized commodities suffer from many shortcomings such as unfair distribution, lengthy distribution process with many leakage opportunities along the way. This paper presents a mobile based solution as an emerging direction in e-Government to tackle this problem. The proposed solution provides transparency, visibility, and control over the distribution process. The solution and its operational process introduce a better mechanism for subsidies distributions. This mechanism improves the effectiveness and efficiency of the distribution process, and ensures high usability. Thus, the needs and competencies of the beneficiaries of the subsidies distribution system are met. The solution also improves the delivery of such an important government service to benefit citizens. Analysis, design, and implementation of such a solution are introduced too.**

Keywords - Mobile wallets; Subsidy distribution systems; Layered Architecture; Design using UML.

I. Introduction

Governments, everywhere, provide consumption subsidies in a number of ways: by providing use of government assets, property, or services at lower than the cost of provision, or by providing economic incentives to purchase or use such goods [1].

In Egypt, traditional means for the distribution of food subsidies suffer from many shortcomings such as: unfair distribution of goods, and lengthy distribution process with many leakage opportunities along the way. One of the conclusions in a recent report published by the World Bank [2] was "*If leakages are eliminated and coverage is narrowed, the government of Egypt could save up to 73 percent of the cost of food subsidies*".

Consequently, better mechanisms for subsidies distributions are needed in order to improve the effectiveness and efficiency of the distribution process and enhance the delivery of such a government service to beneficiary citizens.

The growing rate of subsidies beneficiary population in Egypt has made the development of a fair and efficient subsidies distribution system a real challenge [3]. Moreover, a major sector in the beneficiary population suffers from technology-illiteracy that renders some of the current technological solutions for subsidies management, e.g., smart cards inadequate to scale-up to the magnitude of the problem. Some of these inadequacies are reported in [4]

Currently two subsidy distribution systems are in place. In the first system (an old paper based) a family eligible for subsidized food is given a paper based registry book that carries the family and the monthly ration information. Food is distributed through merchants who are registered with the subsidizing entity to provide this service. A family must choose and register in its registry book one of those merchants based on geographic location.

Each registered merchant has a paper-registry that holds ration information for each family registered with her/him. At the beginning of each month the merchant receives amounts of food for registered families from the subsidizing entity. Families do not have to buy their full rations. At the end of the month cycle, the merchant provides the subsidizing entity with a report containing the amounts of distributed food and the collected money, in return of some percentage as a profit. Amounts of food for the new month are also replenished.

The paper-based system suffers greatly from embezzlement. Merchants may falsely claim larger amounts of food being distributed. This occurs when consumers do not claim their monthly rations and the merchants forge their signatures in the registry. This may also occur through an agreement between the merchant and the eligible family in which case the merchant pays a very low amount of money to this family in exchange of its entire ration.

In the second subsidy distribution system which is currently being deployed after an initial success in pilot regions, each eligible family is issued a smart card containing family and ration information. Each merchant is provided with a Point Of Sale device (POS) and a merchant smart card that contains the merchant ID, the list of families registered with this merchant and the transactions performed for those families. When a consumer acquires a ration, the POS reads both the cards of the merchant and the consumer, and registers the amount of food that the family has acquired on both cards. The rest of the process is similar to that of the paper based except the level of automation introduced. For details please refer to [4].

A. Shortcomings of the Current Subsidy Model

In specific, the current subsidizing models suffer in general from the following main limitations [2] – [4]:

1) The subsidizing entity is isolated from the control of the distribution activity. This isolation leaves the door wide open for corrupted merchants to sell the subsidized goods in the black market.

2) The distribution process is a chain with the subsidizing entity at one end and the target consumers at the other. Along this chain, leakage of subsidized commodities to black market may occur instead of going to eligible consumers.

3) The subsidizing system is complex to be used by various stakeholders. This complexity reflects on the usability of the system by the subsidizing entity as well as the consumer.

4) It is impossible to effectively and efficiently track and report accurate data regarding the current status of the *subsidized commodities.* This reduces the effectiveness of the distribution process, and leaves the subsidizing entity with uncertainty to plan future demand and capacity.

Even with the introduction of the smart card based solution there are still many shortcommings. Connectivity is a main limitation that hinders the application of the system as planned. Because of this limitation, many of the sites are working offline which left the door open for leakage resulting from the possible manipulation of the ration information. Frequent power outage in some areas is another limitation that interrupts the service. Consumers are still tied to a specific merchant; moreover they cannot do multiple transactions during the same month to acquire parts of their ration. The system also does not carry unused balance forward to the next month.

B. Mobile Technology Renders itself as a Solution

In the meantime, mobile technologies have deeply penetrated into the typical life-style of the Egyptian culture at its all population sectors and levels. According to the *ICT indicators in brief* [5], mobile penetration reached around 92% of the population with 27% increase from the year before. This means that mobile based approaches to tackle this problem will have ground success factors.

In this paper, we present a solution based on mobile technologies to tackle the subsidies distribution in Egypt, and for any developing country in general. Accordingly, the main purpose of our solution is to design, develop, and implement a prototype to demonstrate an innovative solution for the subsidies distribution problem using simple and popular mobile phones.

Despite the fact that Smart Mobile Phones gained rapid adoption among mainstream consumer segments across markets [6], we chose to design and implement our system, so that any java-enabled mobile device is sufficient to use the system.

C. Our Solution

Our solution is a type of Mobile government (m-government), which may be defined as [7] "*a strategy and its implementation involving the utilization of all kinds of wireless and mobile technology, services, applications, and devices for improving benefits to the parties involved in e-government including citizens, businesses, and all government units*"

The solution provides the software along with the enabling connectivity to avail the following functions.

1) Enable consumers to acquire and obtain their subsidized rations in smooth way using a mobile device regardless of the device's sophistication (first generation mobile will suffice)

2) Enable Merchants to distribute rations to beneficiaries and have visibility on their accounts using our. This solutin hosts domain software and can be interfaced to variety of terminals (and possibly systems) such as Smart Cards, Mobile devices, and Back-end database server/servers.

3) Enable the subsidizing entity to establish, manage, and control subsidy elements including items, merchants, and consumers, and be able to align these elements based on indicators provided by the system.

This solution intends to transform the entire subsidy distribution process into a technology assisted that is usable by end beneficiaries with minimum investment.

The rest of the paper is organized as follows: In Section 2, we discuss the main requirements that govern the solution starting from a high level usage scenario that shows the main concept of the solution along with the use case modeling of these requirements. The solution architecture with its different views is given in Section 3. Section 4 is dedicated to detailed design summary. Section 5 presents some related design and implementation issues. Section 6 is a discussion that highlights the current status of the solution. We conclude in Section 7.

II. Solution Key Requirements

In this section, we present the key requirements of the solution. We start by introducing a high level usage scenario that abstracts the main idea of the system and is used to derive the system major requirements. We follow by presenting the key system requirements and constraints. Functional requirements of the system modeled as use cases are discussed next.

A. Basic Usage Scenario

To have some high level idea of how the system will support the subsidy distribution process and lay the ground for comprehending the technical discussion that follows, we briefly describe a usage scenario of our solution when it is up and running. In this scenario, we assume that a consumer has her/his own mobile device and is willing to have her/his ration (typically monthly) from a merchant who is previously registered with the system. The consumer has also been registered with the system as well as his monthly quota is known. The scenario takes the following steps, which are also shown on Figure 1 below:

1) The consumer sends her/his id code, merchant code, and choices to the central system.

2) The system verifies the consumer's and the merchant's codes, and balance availability

3) The system sends a transaction number to the consumer and to the merchant

4) Both consumer and merchant should approve the transaction in order for the system to commit the transaction

Fig. 1 shows the common case in which we assume that all consumers have their own mobile devices to post her/his ration request. Our solution takes into consideration a less common case where the consumer does not own a mobile. In this latter case, the consumer may use the merchant mobile device for the same purpose (of course with a secret PIN).

A variation of the device used in executing the scenario is also possible. One possibility is the use of Near Field Communication (NFC) enabled mobile phones [8]. The merchant can also use a PC with a USB mobile modem, a point of sale or his mobile phone. There is still the possibility to use a mobile point of sale, too. We again affirm that any java-enabled mobile device is sufficient to use the system.

B. Reqirements Collection and Constraints

Our approach of collecting user requirements adopted a User Centered Design methodology [9] and has sought to understand user needs from the users directly through observation and user interviews. This user data gathering has focused on the usage of existing (smart card based) and the prior (paper based) systems, and their perception of the important requirements of any future systems. This information has then been used to determine the basic requirements and even assisted in designing an interface proposal for our solution.

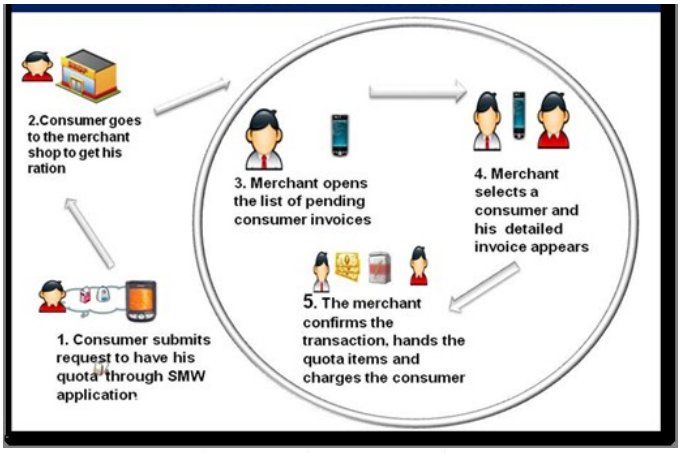
2.Consumer goes to the merchant shop to get his ration
3. Merchant opens the list of pending consumer invoices
4. Merchant selects a consumer and his detailed invoice appears
1. Consumer submits request to have his quota through SMW application
5. The merchant confirms the transaction, hands the quota items and charges the consumer

Figure 1. Basic Usage Scenario.

To understand existing systems and collect the new system requirements w have conducted two major studies. We also benefited from studies done by other researchers in the same respect [3]. The focus of our first study was on the subsidy problem in Egypt with the objective of understanding the main challenges that preclude the implementation of effective and efficient subsidy systems. In addition, the study explored current technological solutions developed to cope with these challenges. The outcome of this study was a comprehensive subsidy study report that exposed many of the proposed system requirements.

Our second study was a large ergonomic study which is concerned with the problems and requirements from a user perspective. In order to understand the user perspective, the work has focused on two classes of user (merchants and consumers). Four regions have been selected that are characterized as having users of varied experience.

One main determining factor which governs our solution is minimizing potential sources of corruption and maximizing the added value for users. A set of basic functional and non-functional requirements [10] must be met that represents the heart of the system. We may view some of these requirements as constraints or assumptions in some sense as given by the following sample. We present functional requirements as use cases (to follow shortly):

1) *System must be mobile-based*

2) *Traditional Mobile devices must suffice to interact with the system at least for consumers.*

3) *A consumer can request all or part of his quota.* This is very different from limitations imposed by the existing system (or the system before) where a consumer is required to request all items of his monthly quota.

4) *A consumer can request quota from any merchant.* This is also very different from limitations imposed by the existing system (or the system before) where a consumer is registered with and tied to one specific merchant.

5) *Consumers own java enabled Mobile to run simple application (or otherwise she/he may use the merchant mobile device)*

6) *Merchant owns java enabled Mobile to run simple application.*

7) *Merchant has some IT skills to run a mobile application*

8) *Customer can respond to simple confirmation messages, and can reply with his pin code.*

In addition to the above basic scenario and the set of constraints/assumptions that represent very basic mandatory requirements, the following section provides discussion about the functional services expressed as a set of the UML use cases [11], [12]. This set of use cases concretizes the basic scenario, and lists the basic services to be provided by the solution. These services achieve the goals of the three main actors of the system: the Consumer, Merchant, and the subsidizing entity; mainly the government in our case. If the system can deliver the services that achieve goals of its actors, then such a system to a large extent meets the functional requirements of the system. Summarization of these services is given in the next section in the form of the system use cases.

C. Use Case Model

As a whole the system comprises twenty three main use cases and three main actors. Use cases included cover the main functionalities (services) to be provided by the solution to its actors. These use cases are grouped functionally as follows:

1) *Consumer use cases:* Use Cases that serve consumers.

2) *Merchant use cases:* Use Cases that serve Merchants.

3) *The Subsidizing entity use cases:* Use Cases that serve the *subsidizing entity*.

4) *Data Maintenance use cases:* Use Cases that provide services to enable building and maintaining the basic data of the system.

5) *Authentication uses cases:* Use Cases that provide authentication and validation services.

Please notice that a use case can be used by more than one actor. An example of such use cases are those related to authentication. Notice also that Data Maintenance Use Cases are mainly used by the subsidizing entity actor for building and maintaining the basic data of the system.

Three main actors are identified: Consumer, Merchant, and the subsidizing entity. There were suggestions to consider mobile operator as a fourth actor but this suggestion is ruled out at this stage as mobile operator has no clear role within the current scope except passing different interactions between the three main actors. Figure 2 provides a brief description for each of the three actors.

Each use case is described using a simple template including commonly used attributes for use cases. In addition to the use case name and actors, the description includes for each use case, the basic scenario, alternative scenarios (if any), Preconditions and Post conditions.

Figure 3 is an example of the *Lookup Consumer Information* use case description following this template. This use case is available to the subsidizing entity actor.

User Class	Description
Consumers	The target group that benefits from the subsidized goods, they have to be registered in the system in order to be able to use the service.
Merchants	Refers to subscribed merchants that have to be registered in the system in order to be able to distribute goods to customers.
subsidizing entity	Is the entity that owns and runs the system and its database; in our case it is the government but it can be NGOs, food banks or any entity willing to manage the distribution of subsidized goods

Figure 2. System Main Actors.

III. Architectural Design

In this section, we discuss the architectural design of our solution from different views [13]. Views to be discussed here are: the logical, style (pattern), and deployment views.

A. Logical View

The system could be simply viewed as 3 main interacting sub-systems linked by a wireless network. The first subsystem is a very simple application hosted by the consumer mobile device. The second is also a simple application that is hosted by the merchant mobile device (or simple point of sale). Most of the business logic and transactional data are within the back end where the system database resides too. Figure 4 shows this high level logical view of the system.

B. System Architectural Pattern

Our solution architecture adopts the well-known layered architectural pattern (style) to organize the solution into a number of layers [14] – [17]. Layered architecture is chosen in order to ensure flexibility and modularity. Organizing software architecture into layers is a very common architectural style used in various industrial systems [18]. The architecture consists of 5 layers in addition to the user interface layer that we do not have space to cover.

1) Terminal Interface Layer (TIL): This layer handles user inputs from various possible user interfaces (e.g., smart card, desktop terminal, cell phone terminal). This layer also converts user requests into commands that can be processed by the Business Logic Layer (BLL). Finally, this layer is also responsible for appropriately formatting the output for display on various possible channels. In the current design, two presentation layers do exist. The first layer resides within thin clients running normal web browser for subsidy control at subsidizing entity side. The second presentation layer is implemented with java to run on merchant and consumer mobile phones.

2) Business Logic Layer (BLL): This layer has the following responsibilities:

a) Processes the business logic commands passed from the TIL layer and composes security requests that will be passed to the Security Layer (SL).

b) Interprets the server responses for the TIL layer.

c) Provides support for application specific business processes and the enforcement of business and data integrity rules.

Lookup Consumer Information
Actor
3rd Party
Basic Flow of Events
1. The actor enters ID of the Consumer he/she wants to look up and click Search. 2. The system retrieves the Consumer information from the database 3. The system displays the information 4. The system allows the actor to print out the report.
Alternative Flows
1. If the Actor wants to see the reports for all the consumers in the database, he/she does not fill in the ID of the Consumer and clicks Search. 2. The system displays the information about all Consumers. 3. The system allows the actor to filter data by location, status,…etc. 4. The system allows the actor to print out the report.
Preconditions
1. actor is logged on to the system 2. actor is authorized to record new consumer 3. Consumer already exists in the database
Post conditions
A report of information about a particular Consumer is displayed /printed.

Figure 3. Lookup Consumer Information Use Case.

3) Translation Layer (TL): This layer is responsible of converting the high level commands which are created in the presentation and business logic layers into a stream of byte arrays to be sent to the communication layer.

4) Security Layer (SL): This layer performs the basic security operations needed in order to keep all commands and their associated data secure. This layer performs encryption/decryption operations, and it filters out unauthorized responses.

5) Wireless Communication Interface Layer (WCIL): This layer is responsible for preparing encrypted commands and data passed by the SL for transmission according to a specific data transmission protocol used by the underlying wireless network backbone.

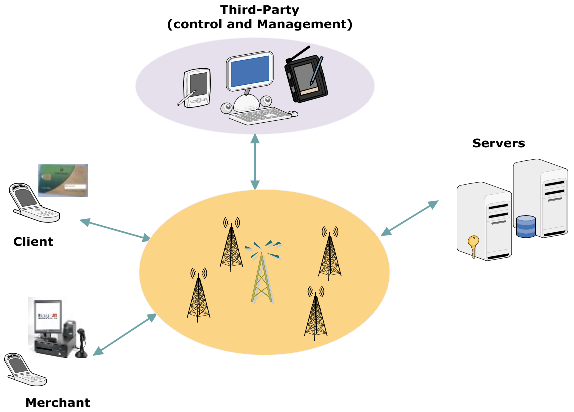

Figure 4. Architecture Logical View.

C. Deployment View

The Deployment View of the solution describes the likely physical network and hardware configurations on which the system will be deployed. This view has been informed by a number of the architecturally significant decisions such as: Centralized Database, Layered Architecture Style, Wireless Network, and Mobile Technology.

The deployment view of the system is represented as UML deployment diagram as shown on Figure 5. Deployment diagrams are large-scale instance diagrams and are important for modeling software architectures in UML [19].

This diagram presents our solution as four UML nodes where each node is connected to the other nodes (mostly) through a wireless network. Each node hosts a subsystem of the solution. The first three subsystems encapsulate the services provided to our principal actors: customer, merchant, and the subsidizing entity.

The fourth subsystem encapsulates the central database that holds subsidy and subsidy-related information along with a large part of the business logic. We may refer to this part as the solution server. Please note that the first and the second nodes abstract mobile devices processing elements that will host software to be used by consumers and merchants. The subsidizing entity may use thin clients or mobile device too.

IV. Detailed Design Summary

Our solution is designed using the common practiced Object Oriented approach where main system entities are perceived as objects. In this respect the detailed design took the following summarized steps:

1) Elaborate business scenarios into more technical (concrete) scenarios.

2) Reformat use cases identified into a set of business operations that represents the minimum system functional commitments.

3) Elaborate business operations identified in 2 above into design elements.

4) Parallel to steps 1-3, develop the class diagram assisted by other UML models (e.g., Sequence and activity diagrams).

5) Decide on persistence and hence lay out the database design

6) Develop user interface

In this section we discuss some of these detailed design activities. For space limitation we are unable to give full details. Details given here are intended for shedding light on the system as a whole. In subsequent papers we dedicate greater details that encompass these activities.

A. Typical Business Scenario from a Design Perspective

In this design activity we elaborate the main business oriented usage scenario into system-wide technical interaction. Compare the following scenario (from a design perspective) to the business scenario shown on Figure 1.

1) The consumer runs the application from her/his mobile device and connects to the back end.

2) The system prompts the consumer to enter her/his ID and PIN code to login, if it is the first time to use the system, the consumer is directed to a registration form.

3) The consumer's information is validated at the server side.

4) Consumer sends a request asking for her/his monthly ration.

5) The server responds at the Merchant side by sending a list of consumer names who wants their rations.

6) The Merchant selects a consumer from the list to service by sending to the server the consumer and his ration data including consumer ID, quota type, and details of required quota items.

7) The server responds with a summary of the interaction to the consumer that includes the consumer and merchant identities as well as quota details (including items, ration's month and year, and the total to be paid by the consumer).

8) When the consumer commits this information, the database is updated with this finalized transaction.

B. Mapping Key Business Operations into Design Elements

Here, use cases are analyzed and grouped into a set of key business operations (*Command* is the corresponding technical term used later) that the solution is committed to provide. Identified business operations are then mapped into design elements using more than one model. For example:

1) UML Sequence and activity diagrams are employed in order to capture timing and sequencing of the various commands and activities for each system operation.

2) Trigger-Action-Response (TAR) Model which is a low level behavior model used to specify each operation in terms

ISBN: 978-1-61208-249-3

of: Command Trigger, Server Actions, and Command Response.

Figure 6 depicts a high-level grouping of the key business operations in the proposed design. As shown in the figure, business operations are grouped into four main categorizes:

- User Account Management operations: These operations deal with the registration of the customers and merchants to the system. The operations also cover various activities related to the management of the created account, such as the change of the user PIN.
- Subsidize-Out operations: These operations present the core of the solution. They cover the activity of claiming subsidized items from a merchant. The Subsidize-Out operations include the customer request, merchant approval, and customer approval processes.

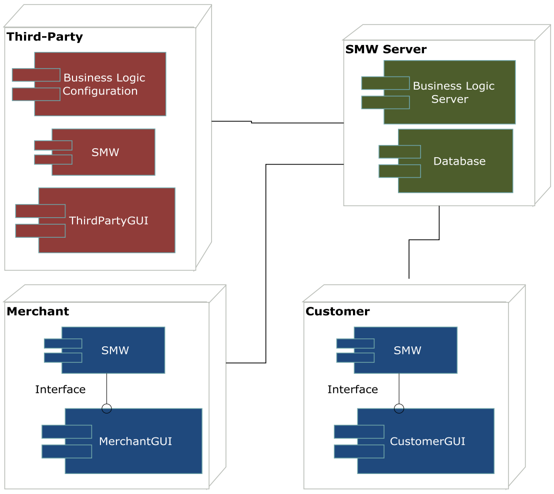

Figure 5. Deployment Diagram.

- Management and Control operations: these operations focus on the activities related to the supply of subsidized items to the merchants and monitoring of the status of these items throughout their life-cycle from their arrival to the merchant until they are claimed by consumers.
- Enquiry-related operations: these operations cover all relevant enquires that are useful for various stakeholders in the system. This includes, for example, operations about enquires related to merchant information, customer available balance, list of customer transactions, etc.

Dealing with each of the leaf business operations as a command, we show on Figure 7 how the command is triggered, and how it climbs up or down the solution layers, and how each layer manipulates the command until its success or failure.

The figure provides an example of how one of the business operations Purchase-Customer request traverses the different solution layers from the moment it is triggered and how it is manipulated along different layers. The role of each layer in processing the command is apparent.

C. Class Diagram

Our class diagram encompasses 2 main categories of classes: business domain classes and utility classes. A major part of the business domain classes are persistent classes too. These include business entity classes such as Items, ration quota, consumer, merchant, and other related classes e.g. organizations, governorates, cities, and subscription. This set of classes forms the core of the system database.

Utility classes span all layers of the system architecture to provide system wide support functionalities. These can further be divided into general utility classes, client interaction classes, and server interaction classes.

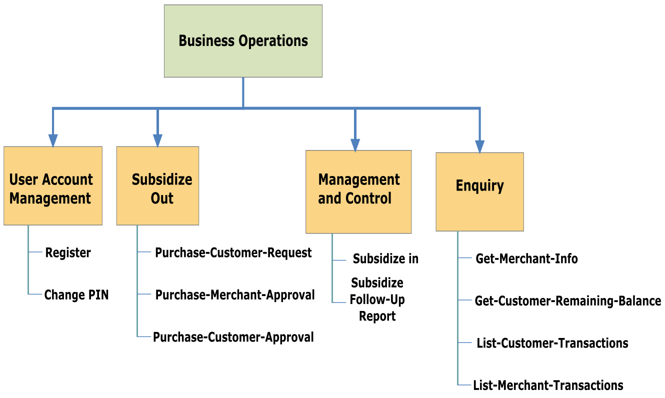
Business Operations
User Account Management
Register
Change PIN
Subsidize Out
Purchase-Customer-Request
Purchase-Merchant-Approval
Purchase-Customer-Approval
Management and Control
Subsidize in
Subsidize Follow-Up Report
Enquiry
Get-Merchant-Info
Get-Customer-Remaining-Balance
List-Customer-Transactions
List-Merchant-Transactions

Figure 6. Key Business Operations.

General Utility Classes model abstractions such as user request/response, Translation class (responsible of converting command data into byte array to be sent to the server), and encoding/decoding classes

Client Interaction Classes are a set of other client classes that work just below the user interface classes to initiate a command (service) based on user entry and send the created command to the translation layer, or receive the response of previously sent command and handle errors if found .

Server Interaction Classes represent database wrappers that shield the server database. Any interaction with the back end database is done through this database wrapper.

V. Design and Implementation Issues

In this section we briefly discuss some issues related to both design and implementation of the solution. This discussion sheds some light on our solution specifics and justifies some technical decision we made.

A. Design Issues

Heavy Use of Inheritance: our class diagram is characterized by heavy use of Inheritance mandated by the nature of the solution (in specific the structure of the utility classes). In addition to being a necessity as we perceive it in our case, heavy use of inheritance in our solution agrees with the common Java programming practice [21]; the language we used to implement the solution.

Assigning Responsibility to classes: in finalizing the class diagram starting from the domain model of the problem concepts, we have employed UML Sequence diagrams along with the basic design patterns. This way we could concretize use case (business operation) interactions in a systematic way and hence assigning responsibilities to different classes. This leads to reduced coupling and increased cohesion of the classes [22]. We also used UML activity diagram to flesh out algorithmic logic for some of the class operations.

Command Triggering

Input from TIL	A customer-purchase command code with parameters: Merchant ID, Item list
BLL Action	Append Mobile phone, Customer PIN and compose the command
Output to SL	A request to encode the composite command

Server Actions

```
If the submitted data are valid
        Generates a new transaction number
        Store the Transaction information in the
        transaction database with transaction status
        marked ‘not-confirmed’.
        Send the transaction number to Customer.
        Send the complete transaction information to
        Merchant
Else
        Send error code to Customer
End if
```

Customer Response

Input from SL	Server response containing transaction number or error code
BLL Action	Store the purchase transaction number if found, On error, interpret the error code
Output to TIL	Trigger the Transaction-Started event, or Trigger the Transaction-Rejected-Error event

Merchant Response (Conditional)

Input from SL	Decrypted server response containing transaction information
BLL Action	Add the purchase transaction information to the list of pending transactions
Output to TIL	Trigger the Pending-Transaction-List-Updated event

Figure 7. Life Time of the *Purchase-Customer* Command.

B. *Implementation Issues*

Java is used as the programming language for coding the system. Java provides all we need to implement the solution. Java is also selected for its native support to network communication. As the reader may have noticed, an important part of the system relies on network communication to send commands and receive responses.

For the back end implementation we are employing Oracle products to host data, business logic, and the part of the application used by the subsidizing entity. Oracle database engine (Oracle 10g) hosts the solution persistent data briefly discussed earlier. The database engine also hosts stored procedures and database triggers that implement the core of the business logic. The part of our application that should be used by the subsidizing entity for maintaining the data and extracting indicators is implemented using Oracle development tools too.

VI. Discussion

Our solution has passed through typical phases of development: planning, analysis, design, coding, and testing interleaved in an iterative way as the recent best practices recommend. We can say that what we have today is a prototype that has been demoed to relevant stakeholders and communities. We received many valuable comments and advices. Many of these comments are business oriented and few are technical. Business comments are centered on extending the solution to cover different spans of subsidies such as subsidies provided by NGOs. Many comments also suggested incorporating other forms of subsidies in our solution such as gasoline.

Examples of technical comments include the need for: more thorough testing and validation, dealing with security issues, performance concerns, and solution usability.

We do not claim that the prototype is ready to be deployed now to a real working environment. Still more work is required to productize the solution. Luckily we have thought of many of the comments and questions raised during the demos. For example we have taken into consideration while designing the system both on the level of the clients and the back end that it can accommodate extensions in different dimensions. The current version of the solution supports other donors of subsidy such as NGOs not just government. It is also true that the solution requires more testing especially performance and usability testing. Concurrency and synchronization issues require more analysis too.

As for the security issue of our solution, we believe that it is too critical to be overlooked. We have a draft pits and pieces of a complete solution that addresses the problem from different angles but it is not complete yet. We also realize that a security solution for our system is complex due to different factors. Many sub-systems form our system including operating systems, databases, web applications, and mobile applications. Many actors are supposed to use the system with different roles. Added to these is the need to address two security aspects that intersect with the above factors, namely secure communication and access control. We have laid out some visionary model that encompasses these different factors and aspects but we did not concretize it yet.

Within this context we still need to evaluate different available options and tradeoffs. For example we are studying the viability of encrypting information while in transit. We also evaluate the level of auditing we should incorporate into the solution and the effect this will have on the system performance. A work is also required to prepare a full access matrix mixed with role-based privilege assignment.

Currently, there is undergoing discussions and negotiations with the key stakeholders of the subsidizing entities to finish the above issues and productize the prototype. This will be followed by deploying the system in a metropolitan area for experimenting with the solution. We will report later about the emerging status of our solution including the security model.

VII. Conclusion and Future Work

In this paper we presented a complete high level picture of the requirements, design, and implementation of consumption subsidy solution that provides automated support for the subsidy distribution process using mobile technology. The application can run on any java-enabled mobile and interacts with a back end server maintained by the subsidizing entity(s). The system provides the level of control and visibility over subsidy distribution process from the side of the subsidizing entity (typically governments) that helps in minimizing leaks to black market. In addition, it makes the process easy for consumers and merchants. For consumers it provides them with new services of requesting his ration from any registered merchant and is no longer tied to a specific one. Also consumers are not required to ration all (monthly) quota at the same time.

Our next steps include productizing the solution by completing the issues of security, performance and usability. More rounds of thorough validation and testing are also among our next steps. The completion of these steps and the deployment of the solution in a metropolitan area for more experimentation with the solution rely largely on the success of the undergoing negotiations with the key stakeholders of the subsidizing entities to sponsor and finance the required work.

Acknowledgment

This work benefited from the valuable contributions of the team members of the project C2/S1/145 of EU - Egypt Innovation Fund sponsored and funded in part by EU. The content of this paper is the sole responsibility of the author and can under no circumstances be regarded as reflecting the position of the European Union.

REFERENCES

[1] http://en.wikipedia.org/wiki/Subsidy. 14.8.2012.

[2] Sherine Al-Shawarby, Heba El-Laithy, Ahmad Iman Youssef, and Iman Sadek, “Egypt’s food subsidies: benefit incidence and leakages,” Social and Economic Development Group, Middle East and North Africa Region, The World Bank, September 16, 2010.

[3] Nagowa ElFaoyal, et al., "Governmental subsidy to goods and services poll to public opinion,” The National Center for Social and Criminal Studies, Cairo, Egypt, 2008.

[4] Magdy Amer, “Using Smart Cards to Control the Distribution of the Subsidized Food in Egypt,” in Proc. 1st International Conference on Innovation and Entrepreneurship, Cairo, Egypt, 23 – 24 April 2012, pp. 22 -25.

[5] “Egypt ICT Indicators in Brief,” released by Egypt Ministry of Communications and Information Technology, Feb 2011.

[6] “2012 Mobile future focus: key insights from 2011 and what they mean for the coming year,” comScore, Inc, Feb. 2012. Accessed at http://www.the-exchange.ca/upload/docs/comScore%202012%20Mobile%20Future%20in%20Focus.pdf. 10.9.2012.

[7] Ibrahim Kushchu, “Mobile government: an emerging direction in e-government,” Mobile Government Consortium International, UK, 2007.

[8] K. Preethi, Anjali Sinha, and Nandini, “Contactless communication through Near Field Communication,” International Journal of Advanced Research in Computer Science and Software Engineering, vol. 2, issue 4, April 2012, pp. 158 - 163

[9] Eeva Kangas and Timo Kinnunen “Applying user-centered design to mobile application development,” Communications of the ACM - Designing for the mobile device, vol. 48, issue 7, July 2005, pp. 55 – 59

[10] Ian Sommerville, Software Engineering. 9th ed., Addison-Wesley, 2011.

[11] Grady Booch, James Rumbaugh and Ivar Jacobson, Unified Modeling Language user guide. 2nd ed. (Addison-Wesley Object Technology Series), Addison-Wesley Professional, 2005.

[12] Jon Whittle, “Precise specification of use case scenarios,” in Proc. the 10th international conference on Fundamental approaches to software engineering (FASE'07), Springer-Verlag, Berlin, Heidelberg, 2007, pp. 170-184.

[13] Nenad Medvidovic et al., “Modeling software architectures in the Unified Modeling Language” ACM Transactions on Software Engineering and Methodology, vol. 11, issue 1, January 2002, pp. 2 – 57.

[14] Jon Oldevik and Oystein Haugen “Architectural Aspects in UML,” *Model Driven Engineering Languages and Systems, 2007*, pp. 301-315.

[15] Fowler Martin, Patterns of Enterprise Application Architecture. Addison-Wesley, 2003.

[16] Valentino Lee, Heather Schneider, and Robbie Schell, Mobile applications: architecture, design, and development. Prentice Hall, 2004.

[17] C. Hofmeister, R. L. Nord, and D. Soni, “Describing software architecture in UML,” in Proc. the 1st Working IFIP Conference on Software Architecture *(WICSA1)*, Kluwer Academic Publishers, Boston, Dordrecht, London, 1999, pp. 145-159.

[18] J. Savolainen and V. Myllärniemi, “Layered architecture revisited - comparison of research and practice,” Joint Working IEEE/IFIP Conf. on Software Architecture 2009 and European Conf. on Software Architecture 2009, WICSA/ECSA 2009, Cambridge, UK, 14-17, September 2009, pp. 317 – 320.

[19] Chris Luer and David S. Rosenblum, **“**UML component diagrams and software architecture - experiences from the WREN Project” 23rd International Conference on Software Engineering, Toronto, Canada, 2001, pp. 79-82.

[20] Deepak Vohra, JDBC 4.0 and Oracle JDeveloper for J2EE development. Paket Publishing, 2008.

[21] Ewan Tempero, James Noble, and Hayden Melton, “How do Java programs use inheritance? an empirical study of inheritance in Java software.*”* in Proc. of the 22nd European Conference on Object-Oriented Programming, Paphos, Cyprus, Jul 2008, pp. 667–691.

[22] Craig Larman, Applying UML and Patterns: an Introduction to Object-oriented Analysis and Design. 3rd ed., Prentice Hall, 2004.