IJCSC International Journal of Computer Science & Communication Vol. 1, No. 2, July-December 2010, pp. 121-127

Determination of Problem Frames Based on Role Activity Diagrams Leading to Function Points : A Case Study

Kavita Agrawal[1], S. K. Bajpai[2] & S. P. Tripathi[3]

[1]Deptt of CSE, BBDNITM, Lucknow, India
[2]Deptt of Comp Sc. & Engg, IET, UPTU, Lucknow, India
[3]Deptt of CSE, IET, UPTU, Lucknow, India
Email: [1]kavitalucknow@gmail.com, [2]skbajpaiiet@hotmail.com, [3]tripathee_sp@yahoo.com

ABSTRACT

To estimate effort at the requirement phase we need to know either lines of code or Function points (FP). Our aim is to indicate that deeper understanding of the structure of a problem, represented through Role activity diagrams and Problem frames, lead to better effort estimation (in terms of FP) at the requirement phase and thus better cost estimation as is desired. Role activity diagrams are first drawn to describe the process flow. These diagrams are then used to determine problem frames which are in turn used for measuring function points.
Keywords: Function Point Calculation, Requirement Specifications.

1. INTRODUCTION

Software development activity is about structure and technique of description. It is important to identify what to describe and how to describe. Effort estimation leading to cost estimation is one activity that is directly influenced by what and how of the description. A deeper understanding of the structure of the problem helps us to estimate cost more accurately at the requirement phase. COCOMO estimation model uses LOC as input to calculate effort whereas COCOMO II model accepts both size in terms of LOC and function points [2],[4],[11],[12]. Lines of code are not known at the requirement phase and can only be estimated based on analogy or previous experience and are therefore approximate. Function points when calculated directly have some drawbacks [2]. For example Function points are subjective in nature not objective as desired. Some research has been done to map problem frames to Function points [7] to make Function points more objective. Problem frames are new and upcoming technique to describe any process and has the potential to become industry standard. To be able to draw Problem Frames efficiently and correctly we need a clear understanding of the system. All its implicit and explicit requirements are to be known and we should be able to describe it in a way useful for the purpose. Some researchers have talked about the usage of Use case points [3]. Some researchers have used role activity diagram (RAD) to describe business models and mapped it into problem frames [5]. This helps us to understand the problem in totality and makes it easy for us to make Problem Frames.

In this paper we use Function point analysis based on RAD that are being mapped to problem frames. Perhaps this point of view may be more useful then as compared to dealing RAD and Problem frames separately. Our aim is to indicate that deeper understanding of the structure of a problem, represented through Role activity diagrams and Problem frames, lead to better effort estimation at the requirement phase and thus better cost estimation as is desired[6]. To further strengthen our view we illustrate our case through an example of Automatic teller machine (ATM).

2. REQUIREMENT STATEMENT FOR ATM

The software to be designed will control an automated teller machine (ATM) having a magnetic stripe reader (card reader) for reading an ATM card, a customer console for interaction with the customer (part of ATM), a slot for depositing envelope (envelope acceptor), a dispenser for cash (cash dispenser), a printer for printing receipts (receipt printer), and a key operated switch to start or stop the machine. ATM will service one customer at a time. A customer will be required to insert an ATM card and enter personal identification number (pin) which will be validated by the bank (networking and bank software is not part of the required software) for each transaction. The card will be retained in the machine until the customer ends the transaction.

ATM must be able to provide following services to the customer:

1. A customer must be able make cash withdrawal from the account liked to the card.

2. A customer must be able to make a deposit to the account linked to the card.
3. A customer must be able to make a balance inquiry of the account linked to the card. A printed receipt can be printed.

A customer should be able to cancel the transaction by pressing a suitable key. Each transaction will be communicated to the bank.

3. ROLE ACTIVITY DIAGRAMS

A RAD has various components, the most common of which are illustrated in Fig. 1 [5]. A RAD may have more than one role. A role can be a human or a software or hardware system. Roles are drawn as set of boxes; activities are boxes within a role. Activities are ordered by states. Interactions are points at which a role interacts with other role to fulfil an objective. Grey activity sends some information to the interacting role. Choices are represented as linked circles. A state may be explicitly marked by concentric circles.

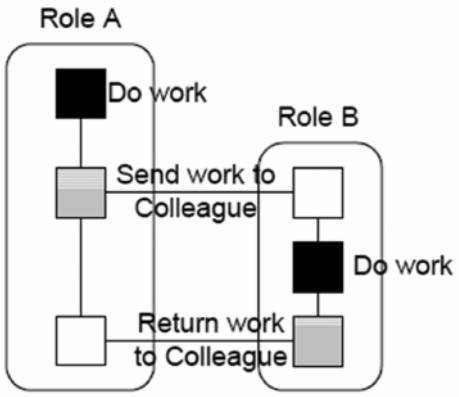

Fig. 1: Elements of a Role Activity Diagram

4. PROBLEM FRAMES

Problem analysis or the Problem Frames approach is an approach to be used when gathering requirements and creating specifications for computer software [8],[9],[10]. User requirements are about relationships in the operational context; not about functions that the software system must perform. The Machine Domain (control machine) represents the piece of software that the customer requires and the platform on which it executes in order to bring about some desired effects.

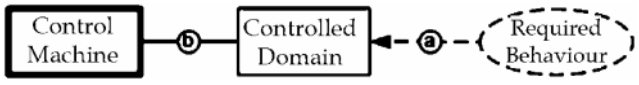

Fig. 2: A Problem Frame

The Problem Domain (controlled domain) is that part of the world in which those effects are perceived by the customer. The Requirements (required behaviour) are the properties that the customer wants to observe, through the shared phenomena *a*, in the Problem Domain as a result of the effects brought about by the software as it executes and interacts with the domain via the shared phenomena *b*.

5. FUNCTION POINT ANALYSIS

Since theory of Function point is well known to the reader, we briefly account the necessary parts in the following [1]. The function point measure is done in three steps:

A. Count and classify the five user function points: external input types, external output types, logical internal files, external interface file types, external inquiry types.Each FP is classified and a Weight is associated with it [1] and Total unadjusted function point (UFP) is calculated.

B. Adjust for processing complexity. The degree of influence of each of 14 general characteristics namely: Data communication, Distributed functions, Performance, Heavily used configuration, Transaction rate, Online data entry, End user efficiency, Online data update, Complex processing, Reusability, Installation ease, Operational ease, Multiple sites and Facilitate change, is taken on a scale of 0 to 5. Where 0 is no influence and 5 is maximum influence. All influences are summed (PC, processing complexity) and an adjustment factor is developed. Where Processing complexity adjustment (PCA)= 0.65+(0.01 * PC) [1].

C. Make the function point calculation.Thus Function Point (FP) = UFP*PCA

6. THE METHODOLOGY

Step 1 : Make RAD then use it to make Problem Frames.

To make a Problem frame we need mainly three things i.e. Domains interacting with the main Machine (main domain), their interactions (interface) and the required behaviour. Some researchers have tried to connect RAD to problem frames to some extent using a case study of global financial services [5].

1. *Domains of Interest:* In problem frames there are domains which correspond to roles in role activity diagrams. One role corresponds to the main machine we want to build other roles correspond to the external domains in the problem frames.
2. *Interface in Problem Frames:* Interface in problem frames correspond to interactions in the role activity diagram.
3. *Required Behaviour:* Required behaviour of problem frames can be derived from actions in

role activity diagram. This gives us an insight on internal logical files and external inquiry.

Step 2: Once we have the Problem frames we need to identify function points. Some researchers have done initial investigations using simple case study of 'sluice gate controller', to provide function point analysis using problem frames. We can find function points under five categories as follows.

1. *External Input Type:* We count each unique user data or user control input that enters the external boundary of the application being measured using Problem frames. Each external input is classified as simple, average and complex depending on number of data elements and internal logical files referenced [1].
2. *External Output Type:* We count each unique data or user control output that leaves the external boundary of the application being measured. Each external output is classified as simple, average and complex depending on number data elements being transformed [1].
3. *External Inquiry:* We count each input output combination, where input produces immediate output. This is classified by taking the greater complexity value of the input and output [1]. In problem frames this is easy to confuse with external input or external output but the required behaviour will indicate external inquiry clearly.
4. *External Interface File Type:* Files passed or shared between domains and is maintained by external domain. In problem frames this can be identified by "a" and required behaviour where it is clearly indicated. Source of input and the information contained in it indicates external interface file.
5. *Logical Internal File Type:* Each major user data and control information group in the machine domain is counted as logical internal file type. In problem frames it forms a part of the required behaviour.

7. THE CASE STUDY

Role activity diagrams of ATM

1. Startup

Operator switches on the machine and checks the cash amount in the machine. He then loads the machine with cash and enters the cash amount loaded in the machine as legitimate cash amount. This information is passed on to the bank and the system is ready to be used.

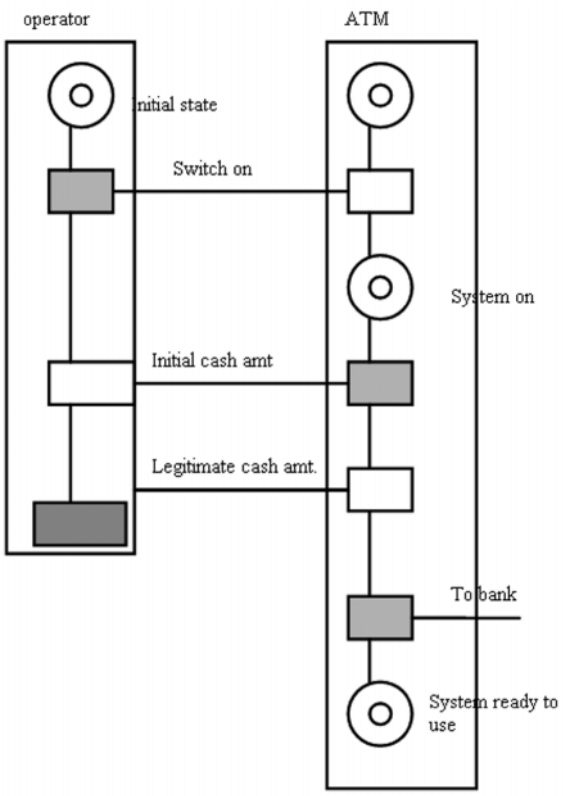

Fig. 3: Start Up

2. Shutdown

Operator checks the status of machine if in use waits till the work is over and then switches it off else if the machine is not in use it is switched off immediately. When machine is switched off it is disconnected from bank and is not available for customer use.

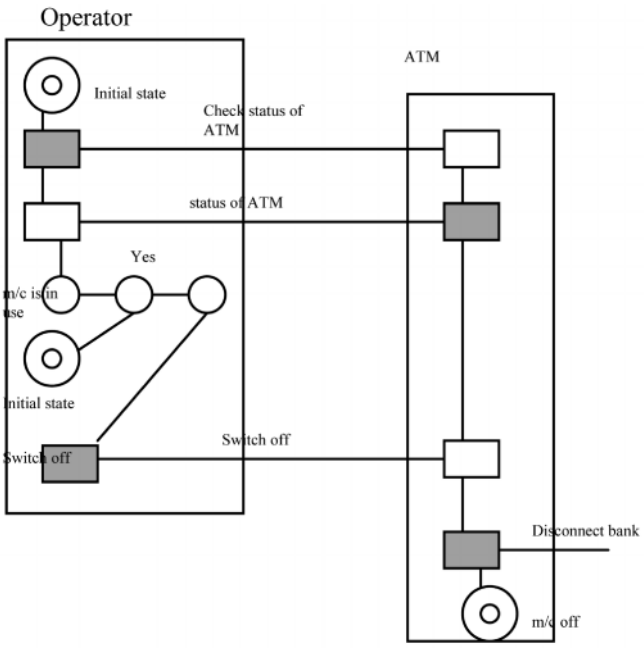

Fig 4: Shutdown

3. Session

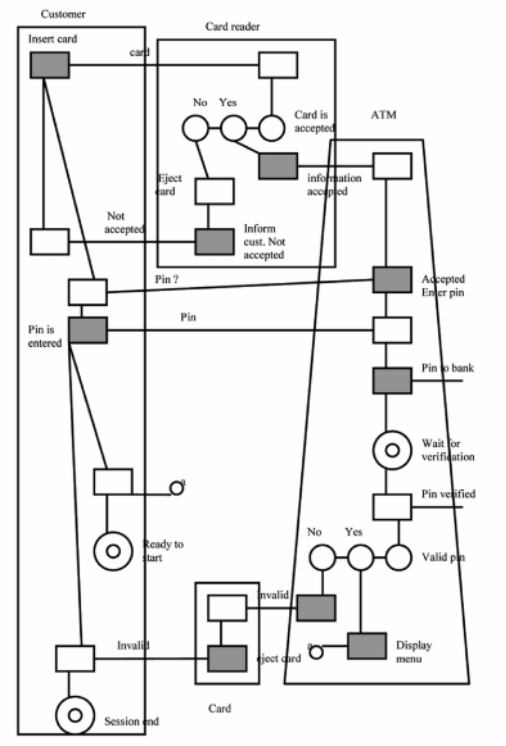

Fig. 5: Session

When customer wants to use machine a session is started. In this customer inserts his card in the card reader. Valid card is accepted else rejected and customer is so informed. Valid card customer is asked to enter PIN. Pin is verified by bank. Authentic pin leads to display of menu with options to withdraw, deposit and inquiry else card is ejected.

4. Withdrawal

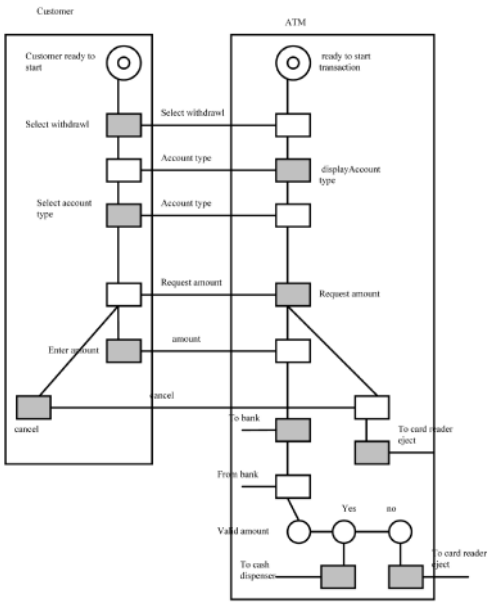

Fig. 6: Withdrawal

From the displayed menu withdrawal option is selected and type of account information is provided to the machine. Next required amount is entered which is checked by bank for valid amount. In case of valid amount cash dispenser dispenses the amount else customer is informed. In between if the customer wants to cancel the transaction he cancels it and the card is ejected out. In the end card of the customer is ejected.

5. Deposit

From the menu displayed deposit is selected and account information is provided to the machine. Amount to be deposited is entered and the amount is kept in an envelope and inserted in envelope acceptor. This information is conveyed to the bank. If customer wants to cancel the transaction he can do so before inserting the envelope. In the end of the transaction card is ejected out.

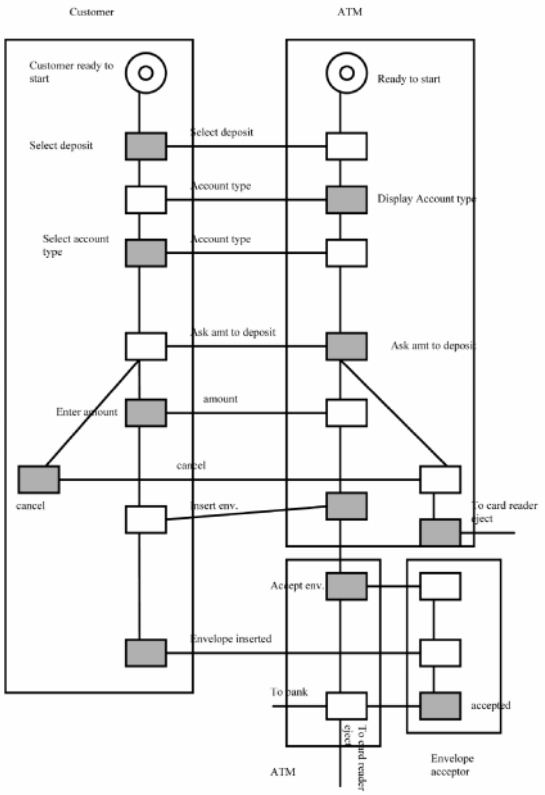

Fig. 7: Deposit

6. Inquiry

From the menu inquiry is selected and account information is entered by the customer. This information is conveyed to the bank and bank responds with the balance amount. This information is passed on to the receipt printer which prints if and the transaction ends. If the customer wants he can cancel the transaction before giving the account information. In the end card is ejected out.

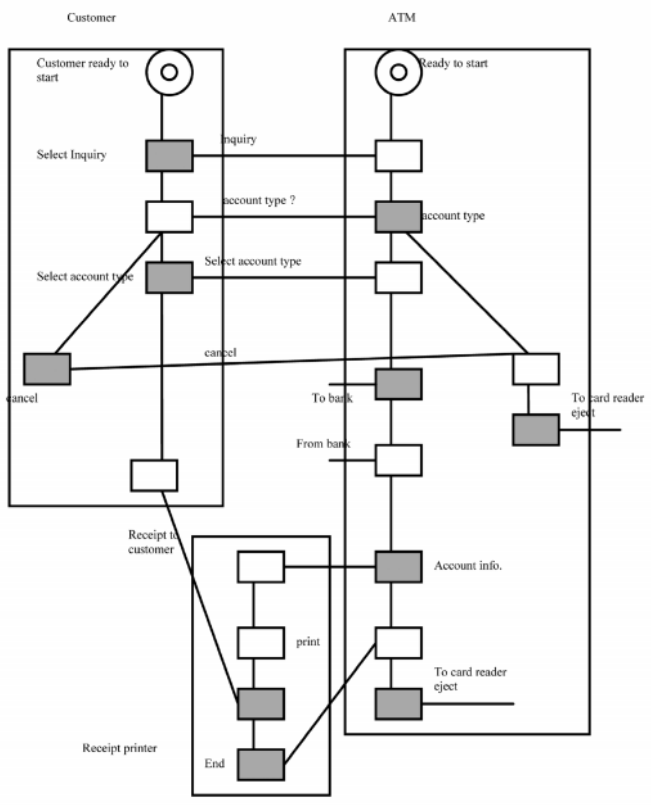

Fig. 8: Inquiry

8. ATM PROBLEM FRAMES AND THEIR FUNCTION POINTS

Problem Frame 1

Fig. 9: Problem Frame 1

Table 1

a: operator to atm	*b: atm to operator*	*c:*	*required behaviour*
Switch on Legitimate cash amount Switch off	initial cash amount	cash balance in the machine	maintain cash balance in the machine

Function points :External file: nil since no data is required to be maintained by the operator. Internal logical files: =1 of low complexity. As the requirement is to maintain cash balance in the machine. External inquiry: nil, External output:=1 of low complexity it corresponds to "b". External input: =1 of low complexity. It corresponds to "a". Switch on and off are are not considered as part of input because it is activating or deactivating the machine after which we start work.

Problem Fame 2

Fig. 10: Problem Frame 2

Table 2

a: bank to atm	*b: atm to bank*	*required behaviour*
pin verified amount validated account information	pin to be verified amount inquiry in withdrawal amount deposit in deposit account information inquiry	connection to bank pin valid then display menu else eject card dispense cash or reject, update info to bankaccept envelope or rejectkeep account info in temporary storage

Function points: External file: =2 of high complexity. It will keep all the information about the customer (2 files because account types considered is 2). Internal logical files: =nil. data sent by bank is kept in temporary variables. External inquiry: nil. External output:=4 of low complexity. Corresponding to "b". External input: =3 of low complexity. Corresponding to "a".

Problem Frame 3

Fig. 11: Problem Frame 3

Table 3

a: card reader to atm	*b: atm to card reader*	*c:*	*required behaviour*
card information	accept or reject card	card inserted card accepted card rejected	check pin through bank validate or reject cardsave card info for the transaction

Function points: External file:=nil. Internal logical files: =1 of complexity low. To save card information. External inquiry: nil. External output:=1 of low complexity. Corresponding to "b". External input: =1of low complexity. Corresponding to "a".

Problem Frame 4

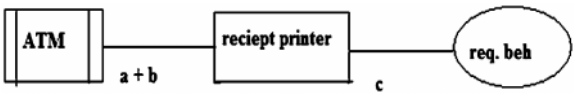

Fig. 12: Problem Frame 4

Table 4

a: receipt printer to atm	*b: atm to receipt printer*	*c:*	*required behaviour*
status of printer	print activated print information	receipt printed	activate printing

Function points : External file:=1 of low complexity. To keep data to be printed. Internal logical files: =nil. External inquiry: =nil. External output:=1 of avg complexity corresponds to "b". External input: =1 of low complexity corresponds to "a".

Problem Frame 5

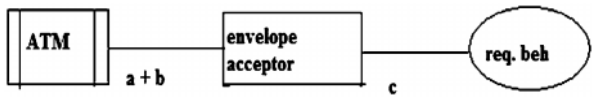

Fig. 13 : Problem Frame 5

Table 5

a: envelope acceptor to atm	*b : atm to envelope acceptor*	*c:*	*required behaviour*
status	activate signal	envelope accepted	activate envelope acceptor

Function points: External file:=1 of low complexity. To maintain status. Internal logical files: =nil. External inquiry: =nil. External output:=1 of low complexity. Corresponds to "b". External input: =1 of low complexity. Corresponds to "a".

Problem Frame 6

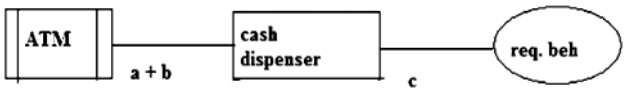

Fig. 14: Problem Frame 6

Table 6

a: cash dispenser to atm	*b: atm to cash dispenser*	*c:*	*required behaviour*
status	activate signal amount to dispense	cash dispensed	cash dispensed

Function points: External file:=1 of low complexity. To maintain status. Internal logical files: =nil. External inquiry: =nil. External output:=1 of low complexity. Corresponds to "b". Both the entries can be treated as one output. External input: = 1 of low complexity. Corresponds to "a".

Problem Frame 7

Table 7

a: cash dispenser to atm	*b: atm to cash dispenser*	*c:*	*required behaviour*
status	activate signal amount to dispense	cash dispensed	cash dispensed

Function points: External file:=1 of low complexity. To maintain status. Internal logical files: =nil. External inquiry: =nil. External output:=1 of low complexity. Corresponds to "b". Both the entries can be treated as one output. External input: = 1 of low complexity. Corresponds to "a".

Problem Frame 8

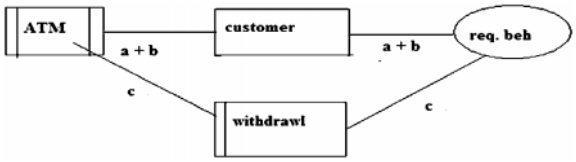

Fig. 16: Problem Frame 8

Table 8

a: customer to atm	*b: atm to customer*	*c: withdrawal has following phenomenon*	*required behaviour*
account type account type amount		sends amount info to bank valid amount activate cash dispenser	request account type request amount activate cash dispenser amount info to bank

Function points: External file:=nil. Internal logical files: =nil. External inquiry: =nil. External output:=1 of low complexity. Corresponds to "b". External input: =2 of low complexity. Corresponds to "a".

Problem Frame 9

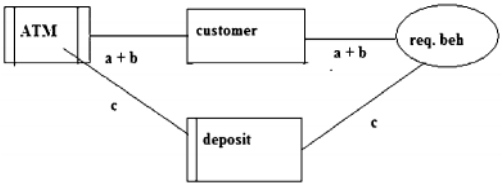

Fig 17: Problem Frame 9

Table 9

a: customer to atm	*b: atm to customer*	*c : deposit has following phenomenon*	*required behaviour*
account type deposit amount	deposit amount	request account type deposit amount send deposit info to bank activate envelope acceptor	account type deposit amount activate envelope acceptor send amount info to bank

Function points: External file:=nil. Internal logical files: =nil. External inquiry: =nil.

External output:=1 of low complexity. Corresponds to "b". External input: =2 of low complexity. Corresponds to "a".

Problem Frame 10

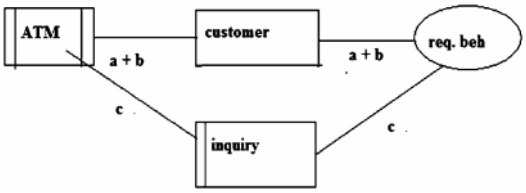

Fig. 18: Problem Frame 10

Table 10

a: customer to atm	*b: atm to customer*	*c: inquiry has following phenomenon*	*required behaviour*
account type is selected	account type	request account type send inquiry to bank	request account type inquiry to bank activate receipt printer

Function points: External file:=nil. Internal logical files: =nil. External inquiry: =1 of complexity low. External output:=nil. Corresponds to "b". External input: =1 of low complexity. Corresponds to "a".

9. CONCLUSION

Cost estimation is not as simple task as it appears to be. With increase in size and complexity this task becomes more difficult. A deeper understanding of the structure of a problem (represented through role activity diagrams and problem frames) leads to better software cost and effort estimation. It is important to get a complete and clear understanding of the problem to get better effort estimation leading to cost estimation. Incomplete understanding at the requirement phase leads to some important points being missed out. In case of Function points, it could result in missing out on few function points and thus leading to inaccurate final calculations.

Some researchers [7] have indicated in their work that function points can easily be found using Problem frames, a new and upcoming technique to describe a system at the requirement phase. Since Problem frames fixes the interfaces, required behaviour and the conditions necessary for them, there are very few chances that two different people will evaluate different values of the function points for the same problem. This strengthens the utility of function points.

Attempts were made to map problem frames directly to function points without the layer of role activity diagrams. It was realized that problem frames are sometimes difficult to identify i.e. more frames are to be discovered [6]. Use of role activity diagrams provides deep understanding of the problem domain, requirements and responsibilities of the machine to be built [5]. This ensures clearer view to the problem and thus problem frames, leading to closer to actual calculations as compared to others proposed so far.

REFERENCES

[1] Allan J. Albrecht and J.E. Gaffney, "Software Function, Source Lines of Code, and Development Effort Prediction: a Software Science Validation", *IEEE Transactions on Software Engineering*, SE-9, **6**, 639-648, 1983.

[2] Chris F Kemerer, "Software Project Management, Readings and Cases" McGraw Hill Company,Irwin Book Team-1997.

[3] E.R.Carroll, "Estimating Software Based on Use Case Points", *OOPSLA'05 -2005*, California, USA ACM. Pg 257-265.

[4] Ian Sommerville, "Software Engineering", 5th Edition, Addison-Wesley.

[5] Karl Cox , Keith Phalp , A. Aurum , S.J. Bleistein , J.Verner, "Connecting Role Activity Diagrams to Problem Frames Approach", *Australian Workshop on Requirement Engineering-2004*, pg 2.1-2.13.

[6] Kavita Agrawal, S.K.Bajpai, S.P. Tripathi, "Application of Role Activity Diagrams in Finding Problem Frames and Determination of Function Points", *ICSTE 2009*, pg 17-20, July 2009 in Chennai India.

[7] L. Lavazza, V.B. Bianco, "Functional Size Measurement Based on Problem Frames; a Case Study", *International Workshop on Applications and Advances in Problem Frames-2008*, Germany pg 44-47 copyright 2008 ACM.

[8] M.A.Jackson, "Software Development Methods", Chapter 13 of A Classical Mind: Essays in Honour of C.A.R.Hoare, A.W.Roscoe", Prentice Hall-1994.

[9] M.A.Jackson, "Problem Analysis Using Small Problem Frames", *South African Computer Journal 22*, Special Issue on WOFACS'98, pg47-60, 1998.

[10] M.Lencastre, J. Araujo, A. Moreira, J.Castro, "Analyzing Crosscutting in the Problem Frames Approach", IWAAPF-2006, pg 59-64.

[11] Richard Fairley, "Software Engineering Concepts", Tata McGraw Hill.

[12] Walker Royce, "Software Project Management A Unified Framework", Addison-Wesley.