Analysis of Transaction Problems using the Problem Frames Approach

Vasantha Banagala
Australian Bureau of Statistics, Canberra, Australia
vasantha.banagala@abs.gov.au

ABSTRACT
A large number of analysis methods are used to analyse and document software problems. Among these, Michael Jackson’s problem frames is an approach to identifying and describing recurring software problems. Data and business transactions are a recurrent software problem pattern in business and commerce systems. This paper demonstrates that problem frames can effectively capture and describe transactional problems. An extended version of the workpiece frame is proposed for this purpose. We found that problem frames can accommodate the critical transactional properties on diagrams and identify the difficulties associated with solving it.

Categories and Subject Descriptors
Information Systems

General Terms: Design

Keywords: Analysis, Problem frames, Transactions, Data transactions, Business transactions, Analysis patterns, Workflows, Context diagram, Workpiece problem

I. ANALYSIS OF BUSINESS INFORMATION SYSTEMS
A large number of analysis methods are used to analyse and document business information systems and related requirements. Analysis is defined as the study of a problem domain, the achievement of understanding of and documentation of the characteristics of that domain and the problems (requiring solution) that exist within that domain. It is also a uniquely human activity (not automated) and is centered upon constructing abstractions and creating links between abstractions [2,11,12].

Analysis of a business information system may typically engage Structured Analysis (SA), Object Oriented Analysis (OOA) [16,17,18,19], Business Process Modeling (BPM) techniques [2,4,5] and data modeling using ER notations. The reader is referred to reference [2] for detailed comparison of these methods including their strengths and weaknesses.

The primary aim of all the analysis methods is to develop a model of the existing system or the solution system and not identification of requirements. Attempts to infer or refer to requirements from these models have not been very successful. The other major problem with these methods is that analyse models and design models tend to overlap resulting in architecture of particular method.

In comparison, the problem frames approach provides a syntax to exemplify ‘externally visible characteristics’ of a system through selection of appropriate domains and interfaces and hence requirements. It is important to note that definition of software requirement by Alan Davis [23] or Michael Jackson [27] is an ‘externally visible characteristic’ of a desired system. Problem frames approach also has strong support to contextualise and describe requirements because it enables to couch requirements to a known frame [7].

Although problem frames is a promising method of analysis there have not been significant applications in industrial projects; in particular business and financial systems. Further, one recent application of problem frames to an industrial project suggests that though one should be able to find the appropriate problem frame for one’s problem, this is not always that straightforward [7]. As such, there have been examples of authors proposing new problem frames to address the specific problem they are faced with, for example [8].

The study presented in this paper relates to e-business and indeed other forms of business too, through specific requirements of data and business transactions. It acknowledges that transactions establish an operational pattern and they inform software requirements in business systems. The recurring pattern of transactions qualifies problem frames as a potential analysis method. The purpose of this paper is to demonstrate application of generic transactional requirements and properties earliest as possible (i.e. during analysis) using problem frames and supplementary notations and identify synchronisation concerns. This key problem frames concepts used are context diagrams, sub problems, sub problem classes, domain classifications and frames. The reader is referred to reference [3] for detailed description of the problem frames method.

Application of problem frames for business information systems have been described in [7,24,25,26] and raise issues of deriving requirements from process models, linking.
business strategy to requirements and integration of problem frames with goal modelling and BPM. Reference [27] addresses the issue of mutual exclusion. Related work on analysis patterns in business systems has focused on more on solution oriented object modelling [28].

II. TRANSACTIONS IN BUSINESS INFORMATION SYSTEMS
While transactions form the basis of data (lexical) domains, it is more important in banking and financial systems for recording of financial dealings. Data domains include databases, files and documents. The user requirements of a banking system may increase in complexity from simple transfer funds between accounts to functionality of an electronic cheque (example 3). Both these fit well into the definition of requirements by Alan Davis or Michael Jackson which is an ‘externally visible characteristic’ of a desired system [21,25].

The key problem that the transaction concept solves is to cope with the subtle and often difficult issues of keeping data consistent even during highly concurrent data accesses and despite all sorts of failures. This is achieved in a generic way that is essentially invisible to the application logic and to application development so that application developers are completely freed from the burden of dealing with such system issues. This is why transaction is an abstract concept and should be an analysis time consideration [4, 6].

Data transaction processing provides management information systems, decision support systems and special purpose information systems with data to help them achieve their goals [5].

A business transaction is an interaction in the real world, usually between an enterprise and a person where something is exchanged. It could involve exchanging money, products, information, requests for service, and so on. Usually some bookkeeping is required to record what happened. Often the bookkeeping is done by a computer for better scalability, reliability, and cost. Business transaction processing ensures the integrity and consistency of a series of activities. For example when an order is entered and payment is received the inventory of the ordered item is appropriately adjusted and goods are delivered. Analysis should reveal the consistency requirements of the sub problems domains and their durability [6].

Although the transactional requirements appear to be simple data entry to records there can be associated complex protocols of operation that involves multiple domains and centered on around accounts, transactions and repositories. For example, the E-cheque involves a protocol between the payer, payee, payer’s bank and the payee’s bank. The problem frames approach offers significant support to capture the interface between domains and is highly suitable for analysis of problems of this nature.

Most software problems of banking, payroll, commerce and human resource systems are transaction problems involving transactional requirements regardless of the perspectives of system, object (entities) or process. Analysis of business problems from a transactional viewpoint enables full capture of requirements. For example if a e-commerce application to sell goods is required and placing orders and paying on line are requirements for transactional completeness there should also be requirements of inventory control and delivery of goods.

III. RESEARCH CONTRIBUTION
This paper proposes application of problem frames to transactional requirements of business and financial information systems to which they have not been previously applied in a consistent and effective manner. It also imposes generic transactional requirements and properties to improve the requirements engineering process.

Hypotheses 1: It is possible to analyse transaction problems using problem frames approach.

Transactions are a recurrent phenomenon in business systems and consist of a number of different activities. Both transaction and sub problem have common properties; Firstly, each is a unit of independent work. Secondly, they both have all or nothing property. The latter is not mandatory for the sub problem but transaction should ensure atomicity.

Hypotheses 2: The workpiece frame is the right choice to represent transactional problems.

If a transaction is to be represented as a typical problem frame, intuition identifies that it is a workpiece problem; firstly it is initiated by a user (except in a required behaviour situation). Secondly, in computerised environment transaction results in a series of recordings in a number of inert entities within the computer. The only difference appears to be that the workpiece usually represents single domain while a transaction concerns a number of them.

It is doubtful whether the information display problem frame or its commanded variation is useful to represent transactional requirements simply because the information display frame involves real world causal domain while transaction problems involve lexical domains such as accounts and records.

A control frame can show multiple interfaces of control if they belong to a single sub problem. Similarly, it is argued that a workpiece problem is suitable for showing multiple sub transactions in a single frame.

Research Problem: How to ensure the generic transactional requirements and properties during analysis?

The primary focus of this paper is to supplement the problem frames syntax to ensure generic transactional requirements and transactional properties. The secondary purpose is to identify related software developments difficulties and opportunities such as types of transactions, workflows, software agents and distribution.
IV. GENERIC TRANSACTIONAL REQUIREMENTS OF COMPUTER BASED SYSTEMS

1. A business transaction requires the execution of multiple operations (in problem frames these are sub problems). For example, purchasing of an item in a department store involves operations of record order, payment and removal of the item from inventory and issue delivery order. Although the requirement is superficially simple the scalability, reliability, and cost associated with a real system turns it into a difficult problem. In particular, scalability will result in concurrency.

2. To scale up for high performance, transactions must execute concurrently. Uncontrolled concurrent transactions can generate wrong answers. In an air travel reservation system and in competing situation for seats it is important to reserve one seat per customer on a first come first served basis.

3. If a transaction runs, it must run in its entirety. In a department store sale, the item should be either exchanged for money or not sold at all. When the failures occur, as they inevitable do, it’s important to avoid partially completed work.

4. Records of transactions, once completed, must be permanent and authoritative. This is often legal requirement, as in financial transactions. Transactions must never be lost.

5. Systems should be expandable to cater to business growth.

6. Systems should be available to support business activity.

7. Systems should be able to connect well in a geographically distributed enterprise [6].

The initial four requirements in this set are analysis time related. The last three requirements are architecture design time related. The perfect analysis tool will allow modeling the former requirements without consideration of the latter. The problem frames approach to analysis models comes close as demonstrated in this paper.

V. CRITICAL TRANSACTIONAL PROPERTIES

Our enhancements to problem frames ensure the properties of Atomicity, Consistency, Isolation and Durability (ACID) of business transactions [6, 14]. The ACID properties of business transactions are defined as follows.

1. Atomicity

Data repositories and business transactions both have to be atomic (all or nothing), meaning that it executes completely or not at all. There must not be any possibility that only part of a transaction program is executed. For example, suppose we have a transaction program that moves $100 from account A to account B. It takes $100 out of account A and adds it to account B. When this runs as a transaction, it has to be atomic—either both or neither of the updates execute.

2. Consistency

A transaction program should maintain the consistency of data repositories. Data repositories integrity constraints as well as the business predicates should be satisfied. Former is a data repositories design issue while the latter is an application software issue. The consistency requirement of moving funds between accounts is that the amount debited from account A should be equal to amount credited to account B (Law of conservation of money).

The consistency property of data transactions problems is one of ensuring the integrity of key constraints. For example when entering the customer details the name or the customer reference number has to be unique. In business transactions it is an issue of ensuring the integrity key constraints and the consistency of participating data repositories. For example in an airline seat reservation problem a seat has to be allocated to the customer while the number of available seats in the airline database has to be reduced by one. In a distributed system consistency ensures integrity of geographically distributed set of databases.

3. Isolation

A set of transactions is isolated if the effect of the system running them is the same as if it ran them one at a time. For example consider the case of two transactions each try to withdraw the last $100 in an account. If both transactions read the account balance before either of them updates it, then both transactions will determine there’s enough money to satisfy their requests, and both will withdraw the last $100. The proper outcome should be only the first transaction withdraws $100 and the second would find an empty account.

4. Durability

Durability ensures that when a transaction completes executing, all of its updates are stored on a type of persistent storage [6,20]

Atomicity, isolation and durability have the same considerations in both data and business transaction problems.

VI. TRANSACTIONS

This paper discusses three classes of transactions using workpiece frame to describe business case studies. They are flat transaction, nested transaction and distributed transaction [13,14]. Further research is required to relate other classes of transactions to problem frames.

VII. WORKFLOWS

A workflow is a collection of tasks organised to accomplish some business activity [15,16]. Each task defines some unit of work to be done. A workflow ties a group of tasks together by specifying execution dependencies and the dataflow between
tasks. Also there are constraints that particular tasks cannot begin until some other tasks end. Correctness and reliability have to be ensured in workflows.

VIII. SOFTWARE AGENTS
Software agent is an autonomous process capable of reacting to, and initiating changes in, its environment, possibly in collaboration with users and other agents.

IX. IDENTIFYING CONCERNS AND DIFFICULTIES
1. Sub problems - Sub problems should ensure atomicity. For example, when the money is paid (for a goods order) the goods are delivered. Further, distinction should be made between data transactions and business transactions. For example, entering periods and ranges (plus patient name and other information) is a data transaction, while crediting funds to an account is a business transaction. Business transaction ensures the business reliability and data consistency and durability. This important differentiation enables use of special techniques make software fully reliable in a business transaction.

2. Typify workpiece domain interactions - The workpiece domains in information systems are recording domains and all interactions with workpieces can be classified as Create, Retrieve, Update or Delete (CRUD) operations.

3. Data - Firstly analysis should distinguish and separate volatile memory (non recoverable data) and persistent memory (recoverable data). For example the period and range domain is persistent. The workpiece frame shows only persistent (durable) data in the workpiece domain.

4. Concurrency - If concurrency of multiple sub problems is revealed during analysis it creates the need to identify shared data.

5. Simple synchronisation (Mutual Exclusion) - If shared data is identified the design can start a synchronisation activity (during operations of CRUD) on the data. For example the period and range domain is shared data. Simple synchronisation without transaction ensures the consistency of a shared resource (domain). Implementation in database environment facilitates implicit synchronisation.

6. Transactional synchronisation - Transactions ensure the ACID properties of multiple domains involved in an atomic activity. While simple synchronisation is sufficient to ensure consistency of a shared resource, it is not sufficient for transaction involving multiple domains. A database integrated transaction management system provides transactional capabilities.

A transactional sub problem, structured using workpiece without interface to the external domains should identify local transaction problem (data or business). On the other hand sub problems that have interface to the external domains may identify distributed and/or long duration transaction problem.

7. Variant frames - Workpiece problems with connection or control variant identify communication to or control of other domains. This is particularly likely to happen in distributed business and administrative problems.

8. Workflows - Problems that have multiple frames should identify workflows. A composite (workflow frame) frame is a series of workpiece problems.

9. Required behavior - The workpiece may also display required behavior. An example of required behavior is, calculate and credit interest to savings account. Required behavior creates potential for software agents.

X. EXAMPLES
The following examples illustrate application of the problem frames approach to business and financial systems related software development problems. These case studies develop diagrams to ensure that the transactional properties (ACID) are illustrated graphically and identify software development difficulties. Analysis of problems starts with workpiece frame; classified as data or business problem and enriched by ensuring the transactional properties.

Example 1: Patient monitoring problem
The purpose of this example is to demonstrate identification of data transaction problems.

A patient monitoring program is required for the intensive care unit (ICU) in a hospital. Each patient is monitored by an analog device which measures factors such as pulse, temperature, blood pressure, and skin resistance. The program reads these factors on a periodic basis (specified for each patient) and stores the factors in a database. For each patient, safe ranges for each factor are also specified by medical staff. If a factor falls outside a patient’s safe range, or if an analog device fails, the nurses’ station is notified [3].

Entering periods and ranges sub problem

![Fig 1 - Entering Periods and Ranges: sub problem diagram]
Entering periods and ranges is a data transaction problem. It is an atomic problem because the sub problem is not decomposable (all or nothing).

There is no business consistency requirement associated with this sub problem. However, there may be data repository integrity constraint requirements associated with the periods and ranges domain and the patient domain, e.g. unique id of the patients mapping to the periods and ranges domain.

The period and ranges domain should be isolated from concurrent access from other sub problems such as monitoring machine. The isolation property is shown by the S in the lower left corner of the periods and ranges domain rectangle. It indicates that the periods and ranges domain is shared data and require synchronisation. If a database is employed as the repository the underlying technology will ensure this requirement. If implementation technologies include files, documents or memory resident objects, then the program has to lock these objects before updating it.

The period and ranges domain is durable (persistent) data. The durability of the periods and ranges domain is shown with double lines. X indicates that the periods and ranges domain is lexical and B indicates that the medical staff domain is biddable.

**Example 2: Banking Example**

The purpose of this example is to demonstrate identification of business transaction problems which require simple or transactional synchronisation. Absence of conditions and execution dependencies disqualify this problem as a workflow.

A software capability is required to provide operations of

- make deposits and withdrawals
- enquire and set the balance
- transfer funds between accounts.

**Crediting Funds to Account sub problem**

Crediting funds to an account is an atomic problem that should either complete or fail totally. It is a business transaction simply because it involves money. The bank staff accepts a certain amount of cash before crediting the funds to the account. This sub problem involves only one domain and there is no need to ensure consistency amongst other domains. Therefore this problem can be interpreted as a recording action and the sub problem is closer to a data transaction problem.

The account domain is shared data and should be isolated from concurrent access from other sub problems. Because of absence of a consistency requirement, isolation can be effected by simple synchronisation without transactions [1]. This is shown with S in the account domain. It is equally acceptable to replace S with T, to ensure isolation.

The durability of the account domain is shown with double lines. The graphical illustration of isolation and durability of domains is useful for the education of stakeholders on data reliability. For example, if a programmer is assigned the task of constructing a program to debit funds he is assured of reliable data when he sees S and the double lines around the account domain.

Durability is a property of domains (shared or not) and concurrency (identifies interaction) is a property of sub problems. Durability can exist without concurrency. For example, off-line batch processing systems utilise data that is consumed by independent sub problems executing one after another. Similarly, concurrency can exist without durability where the data is volatile (held in volatile memory).

**Transfer Funds between Two Accounts sub problem**

Transferring funds from one account to another is an atomic problem. The way to ensure atomicity is to question whether part of the problem can be treated as an independent sub problem. In this case, debiting from one account can be an independent software module but not an independent sub problem.
problem. A transfer fund is a single transaction that should complete fully to meet its transactional obligation.

The consistency requirement stipulates that the credited amount should be equal to the debited amount. In addition to the transactional consistency, the participating domains may also have data integrity constraints.

The isolation in this sub problem is achieved by transaction T. Simple synchronisation of each domain does not achieve the overall consistency of funds. T1 and T2 in the lower left corner indicate shared data synchronised using sub transactions. There are two durable domains in this example; debit account and credit account.

When a single transaction involves multiple persistent domains, it is possible to decompose to multiple sub transactions. In this case, two sub transactions; debit account and credit account. The two sub transactions, T1 and T2 can be implemented using flat transaction model or nested transaction model. Note that the latter model can improve the performance of the solution.

Example 3: Electronic Cheque

The purpose of this example is to demonstrate identification of distributed transaction problem that involves variant frames. The clearing house domain is a frame of communication variant. The analysis that follows treats this problem as a single transaction but if the bank imposes the conditions that successful clearance by clearing house is necessary before payee’s account is credited this problem qualifies as a workflow.

E-cheque delivers confidential, secure, integral, non-repudiation and authentic payment and is suitable for high valued transactions. It is an order to the bank to pay the specified amount of money from the account of the payer to the person named (payee) therein on or after the specified date. The payers and payees can be individuals, companies, government, banks, financial institutions etc. If payer and payee have accounts in different banks the check must be processed through a central clearing house. E-check is an effective payment instrument (compared to cards and EFT) because it can be carried out over internet and email[10].

E-Cheque problem

Figure 4 shows the E-Cheque context diagram. Note that the Payee’s Computer is included in the context because it acts as a mechanism for communication between the Bank and the Payee. There is no direct event-response command from the Payee; i.e. the Payee does not act as an operator.

E-Cheque sub problem

The E-Cheque problem is an atomic, business transaction involving two domains. The payee’s account is local and the payer’s account is remote. The consistency, isolation and durability properties are similar to the transfer funds sub problem.

Because of the involvement of two domains, this sub problem is a transaction (T) that is composed of two sub transactions T1 and T2. These two sub transactions can be flat or nested.

Sub transaction T2 identifies communication with domains of the Clearing House and the Payee's Bank. When communication is required to a remote database and there is bound to be persistent change in the remote database, it identifies a distributed transaction.

When communication (or control) to external domains are identified, it is important to decide upon the protocols of communication (or control). When scalability is high and the performance is important, communication to an external domain can be implemented using a separate thread. Note that the sub problem usually qualifies as a separate process.

The communication with the Payee's computer is via messages. For example, the Payee sends the electronic cheque to the bank, and receives ‘money credited or not’ advice.
**Example 5: Credit Interest to Accounts Example**

The purpose of this example is to demonstrate identification of required behavior problems.

A software capability is required to compute and credit interest to accounts.

**Credit Interest sub problem**

![Diagram](image)

- **n**: Dm! (Update Amount)  ACl (Retrieve Amount)
- **α**: ACl (Account States, Amount etc)
- **Business Rule 1**: Amount = 0 (constraint)

Fig 6 – Compute and Credit interest: problem diagram

Absence of the operator identifies a required behavior problem. This can be implemented as a software agent.

**XI. SUMMARY AND CONCLUSIONS**

According to Michael Jackson, problem analysis takes you from the level of identifying a problem to the level of making the descriptions needed to solve it. The analysis descriptions should include concerns and difficulties.

Successful demonstration of fitting transactional problems to workpiece frame proves the validity of the hypotheses in section III. Further, when a transactional problem is identified and classified as data or business transaction, the transactional properties of the participating domains; namely atomicity, consistency, isolation and durability are applied to illustrate the diagrams.

It is also possible to identify communication to external domains, distributed and long duration transactions, sub transactions, their execution dependencies, workflows and possible implementation techniques using the problem frames approach.

The difference between the frames in this study and the classical workpiece frame is that they show multiple participating domains in a single frame while the classical workpiece frame shows only a single domain.

**ACKNOWLEDGMENT**

I am grateful to Dr. Karl Cox and Steven J. Bleistein of National ICT Australia Ltd. for encouragement.

**REFERENCES**