



BuyIt Prototype Software Requirements Analysis: A C-BASS Component

by Brian R. Schallhom, Wade S. Jernigan,
and Dana L. Ulery

ARL-MR-444

May 1999

19990518 116

Approved for public release; distribution is unlimited.

DTIC QUALITY INSPECTED 4

The findings in this report are not to be construed as an official Department of the **Army** position unless so designated by other authorized documents.

Citation of manufacturer's or trade names does not constitute an official endorsement or approval of the use thereof.

Destroy this report when it is no longer needed. Do not return it to the originator.

Abstract

This document contains the software requirements analysis for a prototype of **BuyIt**. As a component of the Corporate Business Application Software System (C-BASS), this application automates small purchase requests (under \$2,500). The document follows the process of structured analysis, or step-wise refinement of requirements, as applied to the development of **BuyIt**. The “environmental model” includes a high-level system description, followed by a context diagram and a list of events to which the system must respond. The “behavioral model” includes a data flow diagram (DFD) for each of the seven **BuyIt** subsystems. From this representation, the basic functional specifications are derived and represented in structured English (or program design language). The final segment of the document includes a data dictionary.

Table of Contents

	<u>Page</u>
1. Introduction	1
1.1 BuyIt Prototype	1
1.2 Development Plan and Project Schedule	1
1.3 Contents of This Report	1
2. Structured Analysis Overview	2
3. System Overview	3
3.1 Required Functionality	3
3.2 Context Diagram	3
3.3 Event List	4
4. System Requirements	6
5. Functional Specifications	11
5.1 BuyIt Subsystems	11
5.2 Subsystems Data Flow Diagrams	12
5.3 Processing Narration	17
6. Data Dictionary	26
7. References	45
Distribution List	47
Report Documentation Page	49

•

INTENTIONALLY LEFT BLANK.

List of Figures

<u>Figure</u>	<u>Page</u>
1. Context Diagram for BuyIt	5
2. Major Subsystems of BuyIt	12
3. Prepare Purchase Request Process	13
4. Approve Funds Process	14
5. Approvals Process	14
6. Edit Purchase Request Process	15
7. Prepare Purchase Order Process	15
8. Receive Shipment Process	16
9. Inquires Process	16

List of Tables

<u>Table</u>	<u>Page</u>
1. High-Level Requirements for the BuyIt Prototype.....	4

INTENTIONALLY LEFT BLANK.

1. Introduction

BuyIt is a component of the Corporate Business Application Software System (C-BASS) family of applications, an integrated set of Lotus Notes and Web-based software to support U.S. Army Research Laboratory (ARL) electronic **workflow** and task automation. The motivating force behind this project has been **ARL** downsizing and findings put forth in the Business Process Reengineering (BPR) report on the small purchase process. The BPR “To-Be Model: Small Purchase” [1] identifies potential process improvements—some of which require computer **automation**—that would increase productivity of purchasing operations at ARL. Development of a full-production **BuyIt** will proceed in phases, using an incremental, evolutionary approach.

1.1 BuyIt Prototype. The purpose of the **BuyIt** prototype project is to model a secure client/server system that provides for the processing of small purchase requests. This **proof-of-principle** prototype will alleviate some of the risks involved in implementing new technologies used to build the ARL Intranet. The project will also **refine** requirements described in the ARL Business Process Reengineering (BPR) “To-Be Model” [1].

1.2 Development Plan and Project Schedule. The “**BuyIt** Software Development Plan” [2] states a definition of the problem; gives an overview of technical, management, and reliability issues; and provides a detailed project schedule. The report enclosed herein covers the work accomplished to solidify user requirements (drawn **from** existing high-level design documents) and the analytical expansions used to derive a data flow model, a pseudocode representation of processing, and a data dictionary. As stated in the project schedule, the analysis represents a lo-working-day effort and was completed within the stated timeframe of 13 January 1997 to 27 January 1997.

1.3 Contents of This Report. This document presents the results of a structured system analysis used to derive the software requirements for **BuyIt**, starting with the baseline given in the BPR “To-Be Model: Small Purchase” [1]. The body of the report contains five sections.

- *Structured Analysis Overview* - briefly explains the methodology used to extract the software functional specifications.
- *System Overview* - delineates the basic **BuyIt** concept and outlines the high-level requirements.
- *Requirements* - breaks the generic statements into low-level, derived requirements and describes each in detail.
- *Functional Specifications* - discusses the products of the structured analysis (i.e., the data flow diagrams and structured English narrative) for each subsystem of **BuyIt**.
- *Data Dictionary* - lists each of the **BuyIt** data elements, giving a full description and type for the data model.

2. Structured Analysis Overview

Modern software engineering utilizes structured analysis [3] as a powerful methodology for developing system specifications. Through a series of step-wise refinements, detailed delineations of the system's components and their behavior are extracted **from** high-level descriptions of system features and functions. In other words, primary system elements are broken down into progressively more detailed levels of processes, and the data paths between these processes are defined. Three modeling tools facilitate this decomposition: (1) data flow diagrams (**DFDs**), (2) structured English process narratives (pseudocode or program design language [PDL]), and (3) a data dictionary. Accuracy and precision in progressively expanding design definitions are critical to successful system development.

The results of this analytical approach are systematic elaborations of product requirements, typically expressed as two separate models:

- An environmental model that defines the system's interfaces to the outside world. (See section 3, "System Overview.")

- . A behavioral model that defines the internal behavior the system must exhibit in order to deal with the environment. (See section 4, “System Requirements,” and section 5, “Functional Specifications.”)

3. System Overview

The environmental model typically consists of three components: (1) a concise statement of the system’s purpose or required functionality, (2) a context diagram, and (3) an events list. The context diagram is the highest level DFD. It shows the system as a single process, including user interaction and communication with external systems, as well as data flow input and output. The events list creates an index of outside stimuli to which the system responds.

3.1 Required Functionality. As a system, the **BuyIt** prototype provides a secure, automated means for the preparation, routing, approval, tracking, and reporting of small purchase requests. Table 1 lists the high-level requirements for the **BuyIt** prototype and a general description of what each requirement involves.

3.2 Context Diagram. Figure 1 shows the context diagram for the prototype. Each square in the diagram represents an external entity (e.g. users, functional areas, and legacy systems) with which **BuyIt** communicates. The arrows indicate the data that flow into and out of **BuyIt**. A few elements on the DFD need additional explanation. First, the external entity “User” represents all users of the system, and the data flow associated with this box is limited to display of information. Second, all the other external entities (e.g., “Requester” and “Supervisor”), and their corresponding data flow, show the specific information that is passed to **BuyIt** either by that user or by the system. Lastly, the data store EMPLOYEE contains user information such as name, phone number, address, office symbol, etc.

Table 1. High-Level Requirements for the BuyIt Prototype

Requirement	General Description
Security	Provide security measures to prevent unauthorized access to the system and its data and keep authorized users from performing tasks not allowed in their roles.
Purchase request preparation	Provide a means for the requesters and functional users to input/edit relevant information pertaining to a purchase request.
Automated request routing	Automate the process of routing purchase requests to the various functional areas.
Electronic approval	Provide a means for approving officials and functional users to electronically approve or reject a purchase request.
Receive/Accept order	Provide a means for Receiving to notify requesters of shipment arrival and for the purchaser to accept or decline the order.
Request tracking	Allow users to track the status of active purchase requests currently in the system.
Legacy system interface	Implement automated interfaces to SOMARDS and SAACONS legacy systems.
Reporting	Provide users and management with a means for reporting cycle times and costs.

3.3 **Event List.** The following list contains events to which the system must respond.

- Requester initiates new purchase request.
- Requester prepares request.
- Requester corrects rejected request.
- Requester, supervisor, or budget analyst completes fund source.
- Requester or supervisor cancels request.
- Budget analyst certifies fund source.
- SOMARDS certifies and commits funds.
- Special approving official(s) approves request item(s).
- Property book **officer** attaches item tags.
- Property book officer approves request.
- Contracting officer assigns buyer to request.
- SAACONS downloads purchase order information.
- Supervisor approves actual costs.
- Receiving marks shipment as received.

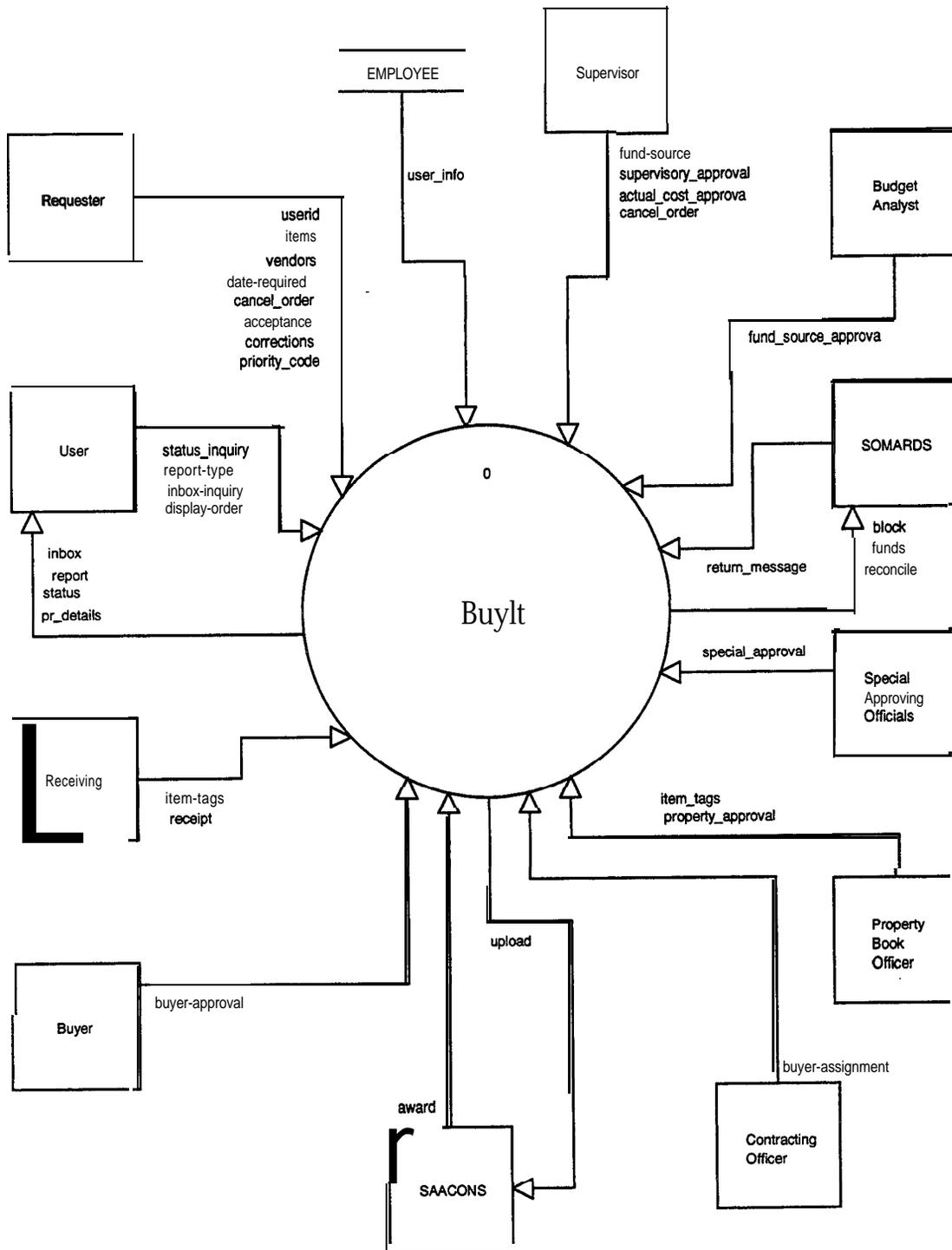


Figure 1. Context Diagram for BuyIt.

- Receiving checks off tagged items.
- Requester accepts shipment.
- User submit status inquiry.
- User request report.

4. System Requirements

Antecedent studies and legacy systems also contribute to **BuyIt's** requirements. The "Automation Requirements" [4] and the "To-Be Model: Small Purchase" [1] were produced during the BPR development effort. However, for some areas, these documents lack detail, and necessary elements had to be derived. Additionally, the limited scope of a prototype necessitated leaving out some of the more complicated or vague automation requirements of the "To-Be Model" (e.g., a pull-down product selection) or areas where the requirement already exists in a Department of Defense (DOD) standard system (e.g., functionalities handled by SAACONS). "BuyIt: Software Development Plan" [2] more fully addresses the constraints on the prototype and the requirements of legacy systems.

E1 Security

E11 Prevent unauthorized access

Description -- Prevent unauthorized access to the system and its data.

Source -- Derived, due to the nature of the system

Interfaces to major functions and external entities:

User.

E12 Enforce role restrictions

Description -- Prevent users **from** performing tasks or accessing/editing data that are out of the scope of their role.

Source -- BPR "To-Be Model" document, Automation Requirements section, requirement A12.

Interfaces to major functions and external entities:

User.
Approvals.
Edits.
Employee address book (for roles).

E2 **Purchase request preparation**

E21 **Create new purchase request**

Description -- Allow the requester to create a new purchase request with preliminary user information filled-in

Source -- BPR "To-Be Model" document, "Automation Requirements" section, requirements A1 11 and A1 14.

Interfaces to major functions and external entities:

User.
security.
Employee address book (for user info).

E22 **Select items**

Description -- Provide a means for the requester to enter item descriptions, specifications, quantities, and estimated costs.

Source -- BPR "To-Be Model" document, Automation Requirements section, requirement A1 12.

Interfaces to major functions and external entities:

User.
Security.

E23 **Complete purchase request**

Description -- Provide a means for the requester and/or approving supervisor to complete the purchase request.

Source -- BPR "To-Be Model" document, "Automation Requirements" section, requirements A1 14.

Interfaces to major functions and external entities:

User.
Security.

E24 Item tag input

Description -- Provide a means for the Property Book Officer to enter item tags.

Source -- BPR “To-Be Model” document, “Automation Requirements” section, requirements A1 13.

Interfaces to major functions and external entities:

User.

Security.

E25 Edit purchase request

Description -- Provide a means for users to edit certain request details as needed.

Source -- Derived, due to the need for making corrections to rejected purchase request.

Interfaces to major functions and external entities:

User.

Security.

E3 Automated routing

Description -- Automate the process of routing purchase requests to the various functional areas and approving officials.

Source -- BPR “To-Be Model” document, “Automation Requirements” section, requirements A12.

Interfaces to major functions and external entities:

security.

Employee address book (for default routing).

E4 Electronic approval

Description -- Provide a means for approving officials and functional users to approve or reject a purchase request.

Source -- BPR “To-Be Model” document, “Automation Requirements” section, requirements A12.

Interfaces to major functions and external entities:

User.

Security.

E5 Receive/Accept order

E51 Receive shipment

Description -- Provide a means for Receiving to notify purchasers of shipment arrival.

Source -- BPR “To-Be Model” document, “Automation Requirements” section, requirements A12. .

Interfaces to major functions and external entities:

User.

Security.

E52

E53 Tag item

Description -- Provide a means for Receiving to attach item tags to received items.

Source -- BPR “To-Be Model” document, “Automation Requirements” section, requirements A3.

Interfaces to major functions and extend entities:

User.

Security.

E54 Accept shipment

Description -- Provide a means for the requester to accept or decline the order or items.

Source -- BPR “To-Be Model” document, “Automation Requirements” section, requirements A3 1.

Interfaces to major functions and external entities:

User.

Security.

E6 Request tracking

Description -- **Allow** users to track the status of active purchase requests currently in the system

Source -- User requested.

Interfaces to major functions and external entities:

User.

E7 Legacy system interfaces

E71 Budget legacy system interface

Description -- Provide an electronic interface to the legacy financial system (SOMARDS) that automates the certification and commitment of funds.

Source -- BPR, “To-Be Model” section, process model diagram A12, process A122.

E72

E73 Procurement legacy system interface

Description -- Provide an electronic interface to the legacy procurement system (SAACONS) that automates the uploading of request information to that system and the download of purchase order data.

Source -- Derived. SAACONS is the primary tool the buyers use during the procurement process; therefore, there is a definite need for interfacing with this legacy system in order to eliminate the need for double entry into both BuyIt and SAACONS.

E8 Reporting

Description -- Provide users and management with a means for reporting cycle times and costs.

Source -- BPR “Reports Specifications” document.

Interfaces to major functions and external entities:

User.

Security.

E9 Navigation

Description -- Provide users with a means for navigating to the various functional areas within the system

Source -- Derived from the requirements previously listed.

Interfaces to major functions and external entities:

User.

5. Functional Specifications

The behavioral model expands the analytical results from the environmental model to more fully define how the system performs prescribed tasks. Typical representations in this model are (1) concise flowcharts showing how information is transformed as it moves through the system and subsystems, (2) a set of structured English statements that form a processing narration based on data types, control structures, and transformations, and (3) a data dictionary defining each data item.

5.1 BuyIt Subsystems. The nine functionalities listed in the previous section identify the major components of **BuyIt**: (1) security, (2) request preparation, (3) automated routing, (4) electronic approval, (5) status tracking, (6) receive/accept notification, (7) legacy system interfaces, (8) report generation, and (9) navigation. These categories consolidate into seven subsystems.

Figure 2 shows the major functional subsystems of **BuyIt**, as represented by a DFD. Each of the seven bubbles in the diagram represents a major subsystem or process of **BuyIt**, with the arrows showing the data flowing into and out of the processes.

The data store **ACTIVE**-located in the center of the diagram--holds all the active purchase requests, each waiting for the appropriate users to perform their functions on them. The **CLOSED** and **CANCELED** data stores contain purchase requests that have been closed out or canceled, respectively. The small squares along the outer edges of this DFD are interfaces to the outside world.

No process bubble for security appears at this level because the application environment (i.e., Lotus Notes) handles unauthorized user security. Additionally, enforcement of role restrictions is handled within each subsystem, as detailed in the following section.

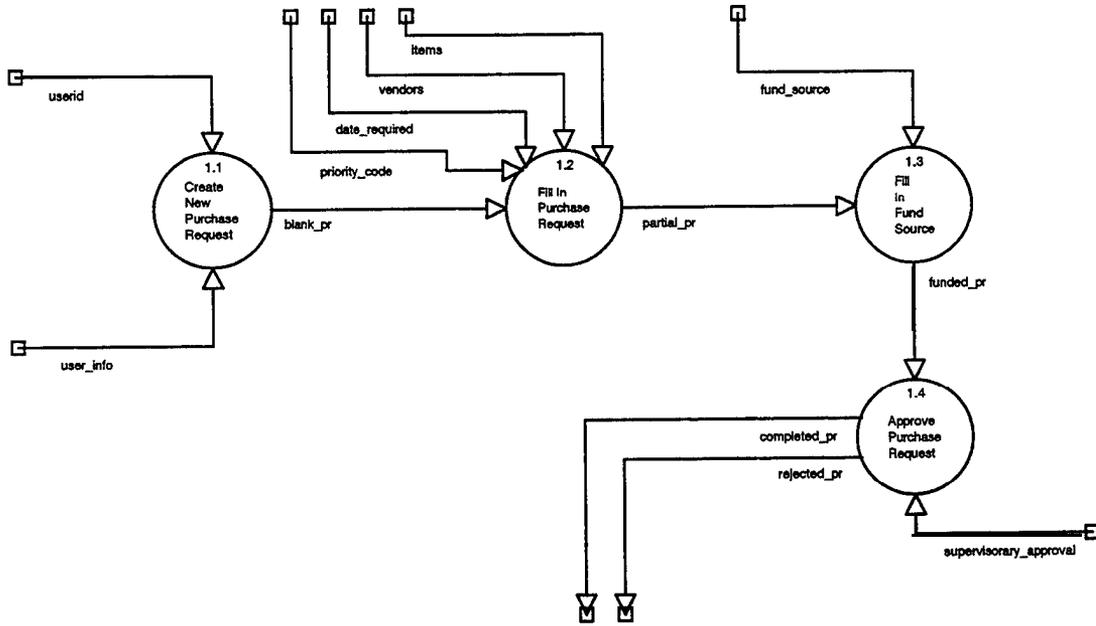


Figure 3. Prepare Purchase Request Process.

at this time), and the date by which the items are required. The fund source completes the information for the request, and the supervisory approval puts the request into the procurement cycle.

Figure 4 shows the DFD for the Approve Funds subsystem. This subsystem, besides having interfaces to users for approvals, also has connections to the SOMARDS legacy system. The Build Block process is executed at the start of the day and creates the transaction block that will be used by BuyIt for the remainder of the day. As certified purchase requests are created during the course of the day, the Certify Funds process queries SOMARDS and grabs the returning message. Depending on the results of this query, the request is either certified or rejected (with explanation). At the end of the day, the Reconcile process is executed to balance the transaction block.

The Approvals process is shown in Figure 5. At this point, the various approving officials attach their approval or rejection to the request. The property book officer also attaches item tags to the individual items in order to flag them during receipt of the shipment.

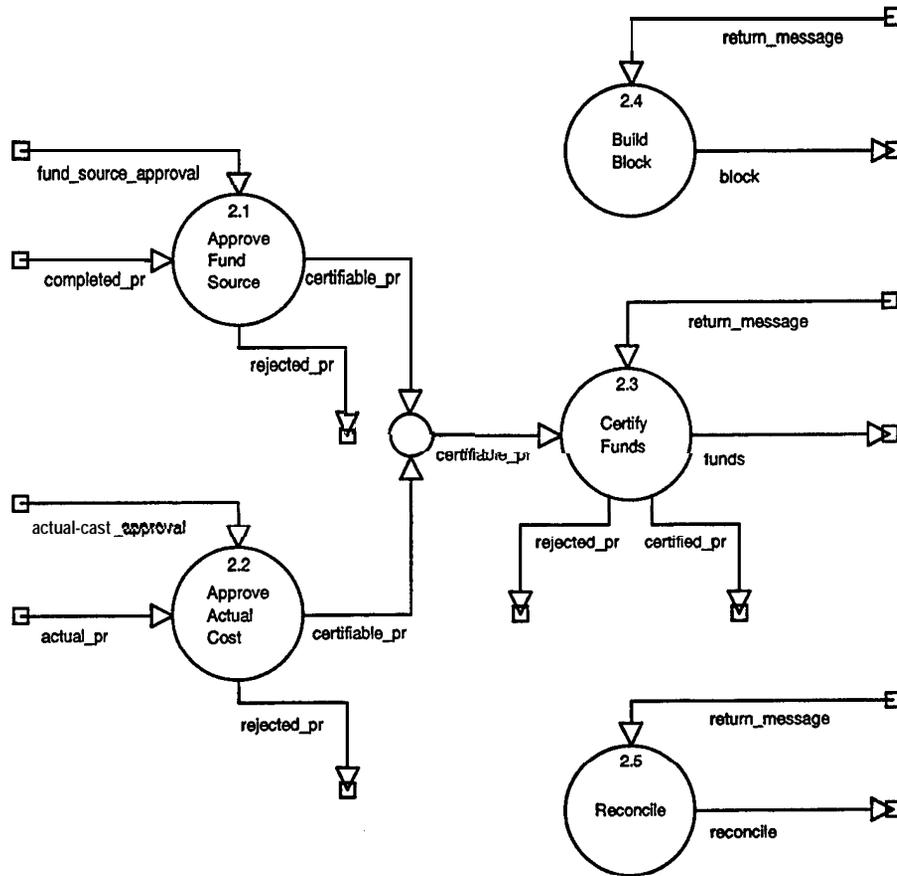


Figure 4. Approve Funds Process.

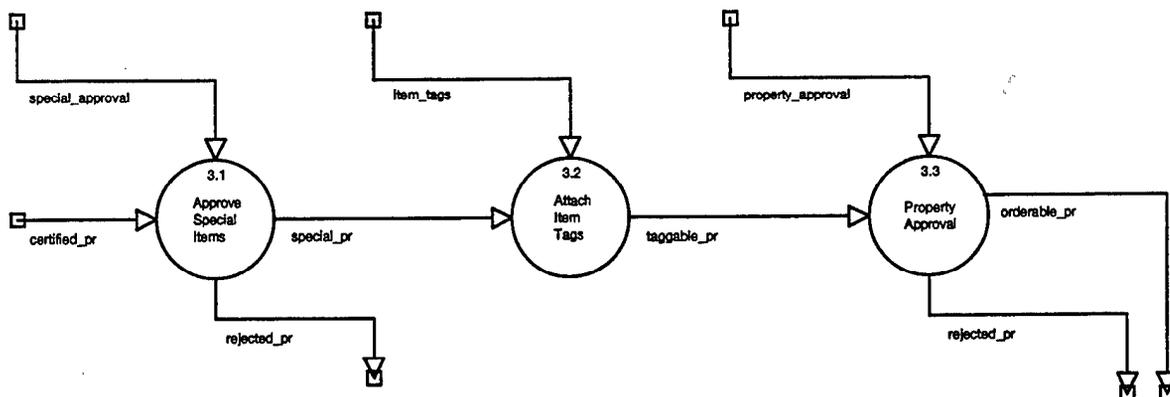


Figure 5. Approvals Process.

Figure 6 diagrams the Edit Purchase Request process. The rejected purchase request is displayed for the requester, who then enters the corrections. What fields within the request the user can edit depends on where the rejection came from and how far along in the approval process the request traveled.

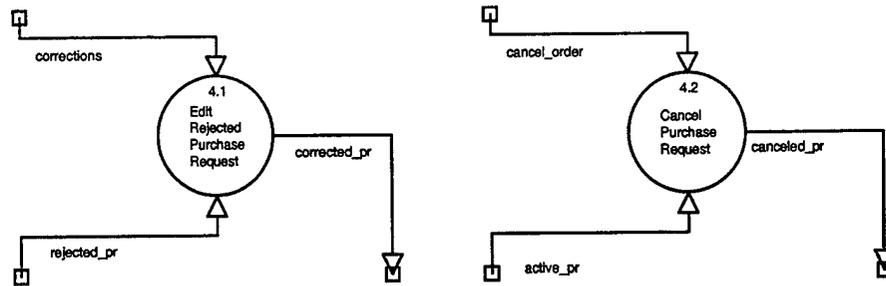


Figure 6. Edit Purchase Request Process.

The Prepare Purchase Order process is diagramed in Figure 7. The first step of the process is assigning a buyer. This triggers the upload of the purchase request information to SAACONS. The buyer then proceeds to work within SAACONS. The second part of the SAACONS interface is the downloading of information (actual costs, selected vendor, delivery date). The request is then forwarded to Receiving in order to alert that office of the pending shipment.

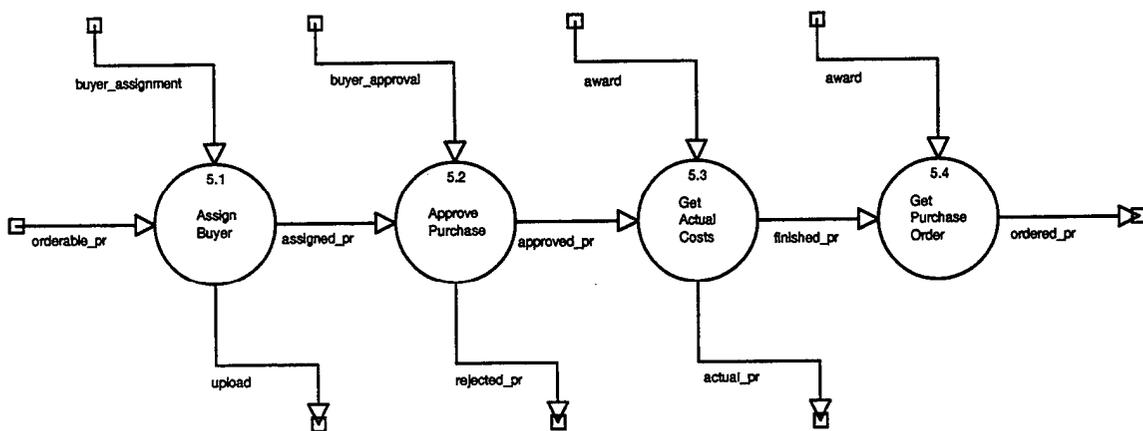


Figure 7. Prepare Purchase Order Process.

Figure 8 represents the Receive Shipment process. Once the shipment has been delivered to the warehouse, Receiving marks the request as received and checks off any tagged items once each has been properly accounted for. At this point, the system **notifies** the requester of receipt and allows the user to accept or return items once they have been delivered.

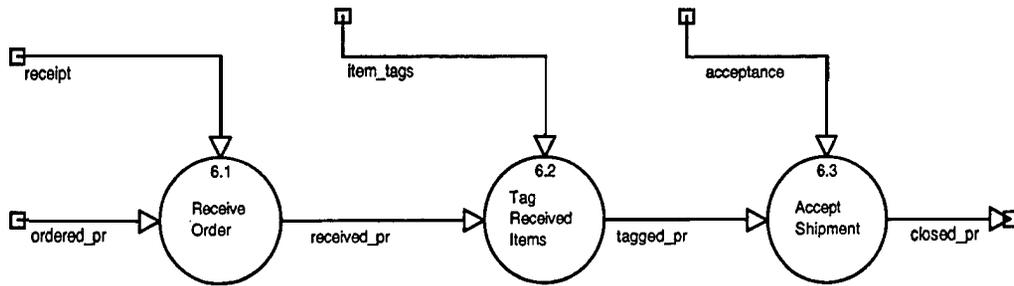


Figure 8. Receive Shipment Process.

The Inquires process is diagramed in Figure 9. The processes shown in this figure are used to display to the user (1) pending actions (inbox), (2) purchase request details, (3) the status of requests, and (4) reports.

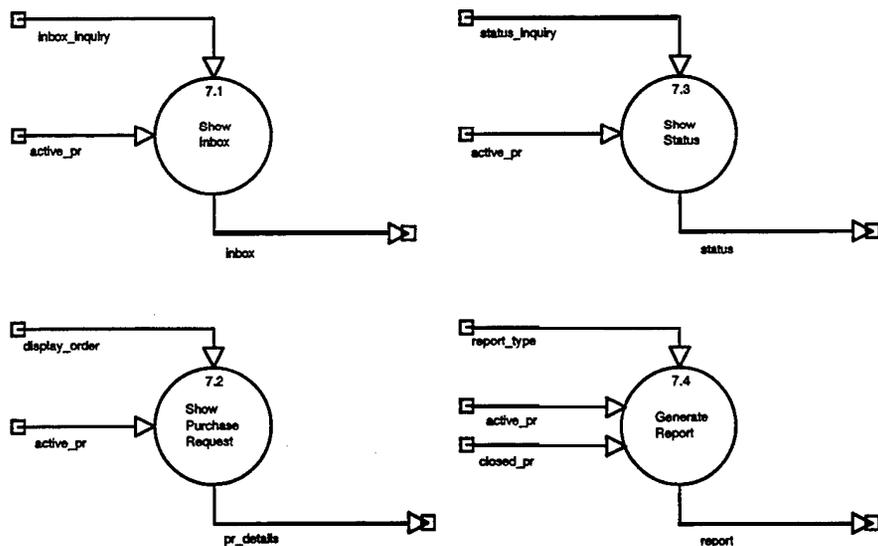


Figure 9. Inquires Process.

5.3 Processing Narration. Having captured the flow of information and identified data objects, each transformation can be further expanded by using the stylized notation of structured English. Basic procedural constructs are combined with English phrases to give concise descriptions for each major operation listed in the prescribed tasks analysis given in section 4.

E1 Security

E11 Login

Input -- userid, passwd

Process:

REPEAT

 GET from user the **userid**, passwd

UNTIL VALID **userid**, passwd

ALLOW login

Output -- N/A

E12 Role

Input -- role, action

Process

GET from EMPLOYEE the role using **requester_userid**

IF VALID action for role THEN

 EXECUTB action

ELSE

 NULL

ENDIF

Output -- action-results

E2 Prepare purchase request for ordering

E21 Create new purchase request

Input -- pr, **requester_userid**, user-info

Process 1 .1

GET from EMPLOYEE the user-info using
requester_userid

SET in pr the requester_userid

SET in pr the user-info using user-info

Output -- blank_pr

E22 Fill in purchase request

Input -- blank_pr, priority-code, items, vendors, date-required

Process 1.2

GET from user the prior&code

SET in pr the priority_code

DO WHILE there is another item to add

 GET from user the item

 SET in pr the item

ENDDO

GET from user the specifications

SET in pr the specifications

DO WHILE there is another vendor to add

 GET from user the vendor

 SET in pr the vendor

ENDDO

GET from user the date-required

SET in pr the date-required

Output -- partial_pr

E23 Fill in fund source

Input -- partial_pr, fund-source

Process 1.3

GET from user the fund-source

SET in pr the fund-source

Output -- funded_pr

E24 Correct purchase request

Input -- rejected_pr, corrections

Process 4.1

DISPLAY to user **rejected_pr** and **explanation**

DO WHILE there are more corrections

GET from user the **corrections**

SET in **pr** the **corrections**

ENDDO

Output -- corrected_pr

E25 Attach item tags

Input -- special_pr, item-tag

Process 3.2

DO WHILE there are more **items** to tag

GET from user **item-tag**

SET in **pr** the **item-tag**

ENDDO

Output -- taggable_pr

E26 Cancel request

Input -- active_pr, cancel_order

Process 4.2

DISPLAY to user the **active_pr**

GET from user the **cancel-order**

PUT canceled_pr into CANCELED

Output -- canceled_pr

E3 Routing

E31 Automated routing

Input -- active_pr

Process -- TBD

Output -- active_pr

E32 Manual routing

Input -- active_pr

Process -- TBD

Output -- active_pr

E33 Assign Buyer

Input -- orderable_pr

Process 5.1

DISPLAY to user the **orderable_pr**

GET **from** user the **buyer-assignment**

SET in **pr** the **buyer-assignment**

SET in **pr** the **inbox_location** to buyer

Output -- assigned_pr

E4 Approvals

E41 Supervisory approval

Input -- funded_pr, supervisory_approval

Process 1.4

DISPLAY to user the **funded_pr**

GET **from** user the **supervisory-approval**

IF **supervisory-approval** is “Yes” **THEN**

SET in **pr** the **supervisory-approval** to “Yes”

SET in **pr** the **request-date** to today’s date

SET in **pr** the **inbox_location** to budget

ELSE

GET **from** user the **explanation**

SET in **pr** the **supervisory_approval** to “No”

SET in **pr** the **explanation**

SET in **pr** the **inbox_location** to requester

ENDIF

Output -- completed_pr, rejected_pr

E42 Fund source approval

Input -- completed_pr, fund_source_approval

Process 2.1

```
DISPLAY to user the completedgr
GET from user the fund_source_approval
IF fund_source_approval is “Yes” THEN
    SET in pr the fund_source_approval to “Yes”
    SET in pr the inbox_location to certification
ELSE
    GET from user the explanation
    SET in pr the fund_source_approval to “No”
    SET in pr the explanation
    SET in pr the inbox_location to requester
ENDIF
```

Output -- certifiable_pr, rejected_pr

E43 Special approval

Input -- certified_pr, special_approval

Process 3.1

```
DISPLAY to user the certified_pr
GET from user the special_approval
IF specialapproval is “Yes” THEN
    SET in pr the special_approval to “Yes”
    SET in pr the inbox_location to
        property_book_officer
ELSE
    GET from user the explanation
    SET in pr the special_approval to “No”
    SET in pr the explanation
    SET in pr the inbox_location to requester
ENDIF
```

Output -- special_pr, rejected_pr

E44 Property approval

Input -- special_pr, property-approval

Process 3.3

```
DISPLAY to user special_pr  
GET from user property-approval  
IF property-approval is “Yes” THEN  
    SET in pr the property-approval to “Yes”  
    SET inbox_location to contracting_officer  
ELSE  
    GET from user explanation  
    SET in pr the property-approval to “No”  
    SET in pr the explanation  
    SET in pr the inbox_location to requester  
ENDIF
```

Output -- orderable-pr, rejected_pr

E45 Actual cost approval

Input -- actual_pr, actual_cost_approval, explanation

Process 2.2

```
DISPLAY to user actual_pr  
GET from user actual_cost_approval  
IF actual_cost_approval is “Yes” THEN  
    SET in pr the actual_cost_approval to “Yes”  
    SET in pr the inbox_location to buyer  
ELSE  
    GET from user explanation  
    SET in pr the actual_cost_approval to “No”  
    SET in pr the explanation  
    SET in pr the inbox_location to requester  
ENDIF
```

Output -- certifiable_pr, rejected_pr

E46 Buyer approval

Input -- assigned_pr, buyer-approval, explanation

Process 2.2

DISPLAY to user **assigned_pr**

GET from user the **buyer-approval**

IF buyer-approval is “Yes” **THEN**

 SET in **pr** the **buyer-approval** to “Yes”

ELSE

 GET from user **explanation**

 SET in **pr** the **buyer-approval** to “No”

 SET in **pr** the **explanation**

 SET in **pr** the **inbox_location** to requester

ENDIF

Output -- approved_pr, rejected_pr

E5 Interface with legacy systems

E51 Build block

Input -- N/A

Process 2.3

PUT to **SOMARDS** the **block**

GET from **SOMARDS** the **return-message**

Output -- N/A

E52 Certify funds

Input -- certifiable_pr

Process 2.4

GET from **certifiablegr** the **funds**

PUT to **SOMARDS** the **block, funds**

GET from **SOMARDS** the **return-message**

IF return-message is “OK” **THEN**

 SET in **pr** the **certification** to “Yes”

 SET in **pr** the **inbox_location** to **special_approval**

ELSE

SET in pr the to certification to “No”

SET in pr the explanation to return-message

SET in pr the inbox_location to requester

ENDIF

Output -- certified_pr, rejected_pr

E53 Reconcile

Input -- N/A

Process 2.5

PUT to SOMARDS the block, reconcile

GET from SOMARDS the return-message

Output -- N/A

E54 Upload to SAACONS

Input -- assigned_pr

Process 5. I

GET from **assigned_pr** the upload

PUT to SAACONS the **upload**

Output -- N/A

E55 Download from SAACONS

Input -- assigned_pr, actual-cost, vendor, delivery-date, po_number

Process 5.3, 5.4

GET from SAACONS the **award**

IF total_actual_cost are greater than

total_estimated_cost THEN

SET in pr the explanation to “Actual cost greater”

SET in pr the inbox-location to supervisor

ENDIF

GET from SAACONS the **award**

SET in pr the vendor, delivery-date, po_number, award-date

SET in pr the inbox_location to receiving

Output -- ordered_pr, actual_pr

E6 Receive shipment

E61 Receive shipment

Input -- ordered_pr, receipt

Process 6.1

DISPLAY to user the **ordered_pr**

GET from user the **receipt**

SET in **pr** the **receipt**

Output -- received_pr

E62

E63 Check off taggable items

Input -- received_pr, item-tag

Process 6.2

DISPLAY to user **received_pr**

FOR EACH item in **received_pr**

IF item-tag is "Yes" **THEN**

 GET from user the **item-tag**

 SET in **pr** the **item-tag**

ENDIF

ENDFOR

SET in **pr** the **inbox_location** to requester

Output -- tagged_pr

E64 Accept shipment

Input -- tagged_pr, acceptance

Process 4.3

DISPLAY to user **tagged_pr**

FOR EACH item in **tagged_pr**

 GET from user the **acceptance**

IF acceptance is "No" **THEN**

 GET from user the **explanation**

 SET in **pr** the **explanation**

ENDIF

```

                                ENDFOR
                                PUT closedgr into CLOSED
                                Output -- closed_pr
E7   Status inquires
                                Input -- active_pr
                                Process 7.3
                                    GET current status from ACTIVE
                                    DISPLAY to user the status
                                Output -- status
E8   Generate reports
                                Input -- report-type
                                Process 7.4 -- TBD
                                Output -- report
E9   Navigation
    E91 Navigate
        Input -- TBD
        Process -- TBD
        Output -- TBD
    E92 Logout
        Input -- N/A
        Process -- TBD
        Output -- N/A

```

6. Data Dictionary

While DFDs and pseudocode (structured English) are important to system specification, additional information is required for a complete analytical model. The content of each data or control item should be more fully identified. A data dictionary is a quasi-formalism for describing content of information as it flows through the system. The standard notation conventions are

Notation

Meaning

=	is composed of
+	and
[II	either/or
{ }n	n repetitions of
()	optional data
* *	comments

BuyIt prototype's data dictionary follows. Each left-hand element is taken from the DFD and the Process Narrative representations of the system. These data items are then given an expanded, unambiguous definition in the right-hand column.

acceptance = *requester accepts or returns shipment item*
["Yes" I "No"]

ACTIVE = (active_pr}

action = **
TBD

action_results = **
TBD

active_pr = *purchase request at some point in the approval cycle*

actual-cost = *actual cost of an item*
units: dollars

actual_cost_approval = *supervisory approval of the total actual cost of the request*
["Yes" I "No"]

actual_pr = *purchase request where the total actual cost is greater than the total estimated cost*
approved_pr + {actual-cost} + {item_tot_act_cost} + total_actual_cost

address = **
street-address + (mail_stop) + city + state + postal_code + (country)

al-cost = *SAACONS line item actual cost*
actual-cost

al_regdeld = *SAACONS delivery date*
delivery-date

approved_pr = *purchase request that has procurement buyer approval*
assigned_pr + buyer-approval

assigned_pr = *purchase request that has been assigned to a buyer by the contracting officer*
orderable-pr + buyer-assignment

award = *SAACONS award download information*
awd_piin + pr_num + spline + awd_status + awd_date + al_regdeld + al-cost + desc_text + vend-name + vend_addrssl + vend-city + vend-stat + vend_zipcode + vend-phone

awd_date = *SAACONS line item award date*
*format: DD-MON-YY”
{legal-character}

awd_piin = *SAACONS purchase order number*
 TBD

awd_status = *SAACONS line item award status*
 {numeric-digit}

award-date = *SAACONS purchase order award date*
 date

bar-code-no = *property book bar code number*
 {alphanumeric}

batch-no = *SOMARDS batch number*
 "BuyIt"

blank_pr = *purchase request with the requester and delivery info
 filled*
 requester_userid + user_info

bldg_no = *building number*
 {alphanumeric}

blk_no = *SOMARDS block number*
 "ARL"

blk_tkt_dt = *SOMARDS block ticket date*
 format: MMDDYY
 date

block = *SOMARDS build block data*
trns_cd + user_auth_key + cmd_dsg + update-code +
blk_no + blk_tkt_dt + tot_blk + batch-no + tot-batch

buyer-approval = *buyer approval of purchase request*
 [“Yes” I “No”]

buyer-assignment = **
buyer_userid + saacons_buyer_code

buyer_userid = **
userid

cancel_order = *order to cancel purchase request*
 [“Yes” I “No”]

CANCELED = { canceled_pr }

canceled_pr = *purchase request that has been canceled*
active_pr + cancel_order

certifiable_pr = *purchase request that has fund source approval or actual
 costs approved*
[completed_pr + fund_source_approval | actual_pr +
actual_cost_approval]

certification = *SOMARDS certification*
 [“Yes” I “No”]

certified_pr = *purchase request that has been certified by **SOMARDS***
certifiable_pr + certification

city = **
{alphanumeric-character}

CLOSED = {closed_pr }

closed_pr = *purchase request that has been accepted by the requester*
tagged_pr + {acceptance}

cmd_dsg = *SOMARDS CMD-DSG*
“1”

company-address = **
address

company_email = **
email_address

company_fax_no = **
phone-no

company_name = **
{legal-character}

company_phone_no = **
phone-no

company_poc = **
name

completed_pr = *purchase request that has been approved by the supervisor*
 @doc_ref_no + funded_pr + request-date + **supervisory_approval**

comt_ref_no = *SOMARDS document reference number*
doc_ref_no

corrected_pr = *purchase request that has been corrected by the requester*
 rejected_pr + corrections

corrections = *corrections to a rejected purchase request*
 TBD

country = **
 {alphabetic-character}

cum_btch_value = *SOMARDS cumulative batch total for the days certification*
 units: dollars

date-required = *date shipment is required by*
 date

delivery-date = *estimated date for delivery from vendor*
 date

desc_text = *SAACONS item description*
 description

description = *item description*
{legal-character}

desc_text = *SAACONS description text*
description

doc_ref_no = *purchase request document reference number*
“W-” + TBD

en-rail-address = **
TBD

EMPLOYEE = {employee }

employee = *employee information - the bare minimum should contain*
user)info + {roles}

eor = *funding element of resource*
(alphanumeric)

estimated_cost = *estimated cost of an item*
units: dollars

explanation = *rejection, cancellation, or return explanation*
{legal-character}

finished_pr = *purchase request where the total actual cost is less than or
equal to the total estimated cost*
approved_pr + {actual_cost} + {item_tot_act_cost} +
total_actual_cost

first_name = *a person's first name*
 {alphabetic-character}

fund_source = **
jo_no + eor

fund_source_approval = *budget analyst approval of fund source*
 ["Yes" | "No"]

funded_pr = *purchase request with a fund source*
 partial_pr + **fund_source**

funds = *funding information for SOMARDS certification*
trns_cd + **user_auth_key** + **cmd_dsg** + update-code +
blk_no + **blk_tkt_dt** + batch-no + **rej_rept_director** +
doc_ref_no + **jo_no** + eor + **act_amt**

inbox = *purchase requests requiring action **from** user*
 {active_pr)

inbox-inquiry = **
 TBD

inbox_location = *current purchase request location*
 TBD

i t e m = **
@line_item_no + description + **unit_of_issue** + qty +
 estimated_cost + actual-cost + item-tag + acceptance +
 item_tot_est_cost + item_tot_act_cost

items = **
{item} + specifications

item-tag = *property book officer inputs “yes” item is taggable,
receiving overwrites with bar_code_no*
[“Yes” | bar_code_no]

item-tags = {item-tags}

item_tot_act_cost = *line item total actual cost*
units: dollars

item_tot_est_cost = *line item total estimated cost*
units: dollars

jo_no = *funding job number*
{alphanumeric}

last-name = *a person’s last name*
{alphabetic-character}

line_item_no = *line item number*
{numeric-digit}

mail-stop = *mail stop or department*
{legal-character}

name = **
first_name + last-name

nomenclature = *SOMARDS certification comment field*
{alphanumeric}

office_symbol = *ARL office symbol*
 “AMSRL-” + (alphabetic-character) + “-” +
 (alphabetic-character }

orderable_pr = *purchase request that can be ordered by procurement*
 taggable_pr + property-approval

ordered_pr = *purchase request that has been ordered*
 finished_pr + delivery-date + vendor + po_number

partial_pr = *purchase request with items and vendors filled in*
 blank_pr + date-required + priority-code + items + vendors

phone-no = *a phone number*
 {numeric-digit}

po_number = *purchase order number*
 TBD

postal-code = *postal/zip code*
 {numeric-digit}

pr_authamt = *SAACONS authorized amount*
 total_estimated_cost

pr_authby = *SAACONS authorized by*
 TBD

pr_buyerid = *SAACONS buyer code*
 saacons_buyer_code

pr_contact = *SAACONS purchase request point of contact*
requester-name

pr_details = *details about the purchase request*
TBD

pr_item = *SAACONS purchase request item details*
pr_num + spline + splum + splcost + splqty + splreqdeld +
desc_text

pr_num = *SAACONS purchase request number*
doc_ref_no

pr_phone = *SAACONS POC phone number*
requester_phone_no

pr_priority = *SAACONS priority code*
priority-code

pr_reqdeld = *SAACONS required delivery date*
date_required

pr_type = *SAACONS purchase request type*
“S”

priority-code = *request priority code*
range:01 -15, 99
{numeric-digit}

property-approval = *property book officer approval*
[“Yes” I “No”]

qty = *quantity requested*
 (numeric-digit }

receipt = *shipment receipt*
 [“Yes” I “No”]

received_pr = *purchase request that has been received*
 ordered_pr + receipt

reconcile = *end of the day SOMARDS reconcile info*
trns_cd + user_auth_key + cmd_dsg + update-code +
blk_no + blk_tkt_dt + batch-no + tot_blk + tot-batch +
ty_act_cd + cum_btch_value + variance

rejected_pr = *purchase request that has been rejected*
 active_pr + explanation

rej_rept_director = “SOMARDS REJ-REPT-DIRECTOR”
“R”

report = **
 TBD

report-type = *type of report to generate*
TBD

request-date = *date purchase request was approved by supervisor*
 date

requester_userid = **
userid

return-message = *message returned from SOMARDS process*
 ["processing complete" | "bad user_auth_key" | "wrong
 update code" | "blk_no/blk_tkt_dt already exists" |
 "accounting class displayed" | "blk_no/blk_tkt_dt doesn't
 exist" | "invalid jo_no" | "invalid eor" | "insufficient funds" |
 "duplicate comt_ref_no" | "cum_btch_value" | "make
 changes" | "variance"]

role = *user role*
 {alphanumeric }

room-no = *room number*
 {alphanumeric}

saacons_buyer_code = *buyer's SAACONS buyer code*
 TBD

sole_source_just = *justification for using a single vendor*
 {legal-character}

special-approval = *approval from a special approving officials*
 ["Yes" | "No"]

special_pr = *purchase request with special approvals*
 certified_pr + special-approval

splcost = *SAACONS estimated item cost*
 estimated_cost

spline = *SAACONS line item number*
 line_item_no

splum = *SAACONS unit of measure*
unit_of_measure

splqty = *SAACONS quantity requested*
qty

splreqdeld = *SAACONS item required delivery date*
date_required

state = *state or province*
{legal-character }

status = **
TBD

status_inquiry = **
TBD

street-address = **
{legal-character}

supervisory_approval = *approval **from** supervisor*
[“Yes” I “No”]

taggable_pr = *purchase request that has had item tags attached by
property book officer*
special_pr + item-tags

tagged_pr = *purchase request that has been received and **taggable** items
have been appropriately tagged*
received_pr + item-tags

total_actual_cost = *the total actual cost of the purchase request*
units: dollars

total_estimated_cost = *the total estimated cost of the purchase request*
units: dollars

tot-batch = *SOMARDS batch number*
units: dollars
["0.00" | cum_btch_value]

tot_blk = *SOMARDS total block*
units: dollars
["0.00" | cum_btch_value]

trns_cd = *SOMARDS transaction code*
["003" | "004" | "3 1 0"]

ty_act_cd = *SOMARDS action code*
"C"

unit_of_issue = **
TBD

update-code = *SOMARDS update code*
["CM" | "NM"]

upload =

user_auth_key = ***SOMARDS** user authorization key* ,
 {alphanumeric}

user_info = *user information*
 name + office_symbol + phone-no + **bdg_no** + room-no

userid = **
 {alphanumeric }

variance = ***SOMARDS** **variance** between tot-batch and
 cum_btch_value - should be **0.00***
 units: dollars

vend_addrss 1 = ***SAACONS** vendor address*
 street_address

vend-city = ***SAACONS** vendor city*
 city

vend-name = ***SAACONS** vendor name*
 company_name

vend-phone- ***SAACONS** vendor phone number*
 company_phone_no

vend-stat = ***SAACONS** vendor state*
 state

vend_zipcode = ***SAACONS** vendor zip code*
 postal_code

vendor = *vendor information*
company_name + company_address + company_phone_no
+ company_fax_no + company_poc + company_email

vendors = *up to three suggested vendors*
(vendor) + sole_source_justification

INTENTIONALLY LEFT BLANK.

7. References

1. Business Process Reengineering Office. ***To-Be Model: Small Purchase***. U.S. Army Research Laboratory, Aberdeen Proving Ground, MD, July 1995.
2. Ulery, D., B. Schallhorn, and W. Jernigan. ***BuyIt: Software Development Plan***. SPAS-97-01, U.S. Army Research Laboratory, Aberdeen Proving Ground, MD, to be published.
3. Yourdon, E. ***Modern Structured Analysis***. Englewood Cliffs, NJ: Yourdon Press, 1989.
4. Business Process Reengineering Office. ***ARL Small Purchase System: Report Specifications***. Draft, U.S. Army Research Laboratory, Aberdeen Proving Ground, MD, October 1996.

INTENTIONALLY LEFT BLANK.

NO. OF
COPIES ORGANIZATION

- 2 DEFENSE TECHNICAL
INFORMATION CENTER
DTIC DDA
8725 JOHN J **KINGMAN** RD
STE 0944
FT BELVOIR VA 22060-62 18
- 1 HQDA
DAMOFDQ
D SCHMIDT
400 ARMY PENTAGON
WASHINGTON DC 203 1o-0460
- 1 OSD
OUSD(A&T)/ODDDR&E(R)
RJTREW
THE PENTAGON
WASHINGTON DC 20301-7100
- 1 DPTY CG FOR RDE HQ
us **ARMY MATERIEL CMD**
AMCRD
MG CALDWELL
5001 EISENHOWER **AVE**
ALEXANDRIA VA 22333-0001
- 1 INST FOR ADVNCD TCHNLGY
THE UNIV OF TEXAS AT AUSTIN
PO BOX 202797
AUSTIN TX 78720-2797
- 1 DARPA
B **KASPAR**
3701 N FAIRFAX DR
ARLINGTON VA 22203-1714
- 1 NAVAL SURFACE WARFARE CTR
CODE B07 J **PENNELLA**
17320 DAHLGREN RD
BLDG 1470 RM 1101
DAHLGREN VA 22448-5 100
- 1 US MILITARY ACADEMY
MATH **SCI** CTR OF EXCELLENCE
DEPT OF MATHEMATICAL **SCI**
MAJ M D PHILLIPS
THAYERHALL
WEST **POINT** NY 10996-1786

NO. OF
COPIES ORGANIZATION

- 1 DIRECTOR
US ARMY RESEARCH LAB
AMSRL D
RWWHALIN
2800 POWDER **MILL** RD
ADELPHJ MD 20783-1 145
- 1 DIRECTOR
US ARMY RESEARCH LAB
AMSRL DD
J J ROCCHIO
2800 POWDER **MILL** RD
ADELPHJ MD 20783-1 145
- 1 DIRECTOR
US ARMY RESEARCH LAB
AMSRL CS AS (RECORDS **MGMT**)
2800 POWDER **MILL** RD
ADELPHJ MD 20783-1 145
- 3 DIRECTOR
US ARMY RESEARCH LAB
AMSRL **CI** LL
2800 POWDER **MILL** RD
ADELPHJ MD 20783-1145
- ABERDEEN PROVING GROUND
- 4 DIR USARL
AMSRL **CI** LP (305)

NO. OF
COPIES IN IZATION

4 DIR USARL
AMRSL CI W **JERNIGAN**
AMSRL CI E W MCCOY
AMSRL CI E B **SCHALLHORN**
AMSRL CI A R ROSEN
2800 POWDER MILL RD
ADELPHI MD 20783-1145

1 DIRUSARL
AMSRL CS COL K **LOGAN**
2800 POWDER MILL RD
ADELPHI MD 20783-1145

1 DIRUSARL
AMSRL SE S A **SINDORIS**
2800 POWDER MILL RD
ADELPHI MD 20783-1 145

1 DIR USARL
AMSRL HR SF B FONOROFF
2800 POWDER MILL RD
ADELPHI MD 20783-1 145

1 DIR **USARL**
AMSRL SL EA P STAY
WSMR NM 88002-5501

ABERDEEN PROVING GROUND

17 DIR USARL
AMSRL HR SF B AMREIN
AMSRLWMMJGRILLS
AMSRL WM **IM** R MCGEE
AMSRLCSAF'DORE
AMSRL **CI** c **NIETUBICZ**
AMSRL CI I D ULERY (12 CPS)

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington Headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188), Washington, DC 20503.

1. AGENCY USE ONLY (Leave blank)		2. REPORT DATE May 1999	3. REPORT TYPE AND DATES COVERED Final, January 1997 - March 1997	
4. TITLE AND SUBTITLE BuyIt Prototype Software Requirements Analysis: A C-BASS Component			5. FUNDING NUMBERS 9UA4EC ACE00	
6. AUTHOR(S) Brian R. Schallhom, Wade S. Jemigan, and Dana L. Ulery				
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: AMSRL-CI-E Aberdeen Proving Ground, MD 210055067			8. PERFORMING ORGANIZATION REPORT NUMBER ARL-MR-444	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)			10. SPONSORING/MONITORING AGENCY REPORT NUMBER	
11. SUPPLEMENTARY NOTES				
12a. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			12b. DISTRIBUTION CODE	
13. ABSTRACT (Maximum 200 words) This document contains the software requirements analysis for a prototype of BuyIt . As a component of the Corporate Business Application Software System (C-BASS), this application automates small purchase requests (under \$2,500). The document follows the process of structured analysis, or step-wise refinement of requirements, as applied to the development of BuyIt . The "environmental model" includes a high-level system description, followed by a context diagram and a list of events to which the system must respond. The "behavioral model" includes a data flow diagram (DFD) for each of the seven BuyIt subsystems. From this representation, the basic functional specifications are derived and represented in structured English (or program design language). The final segment of the document includes a data dictionary.				
14. SUBJECT TERMS BuyIt , software requirements, BASS, C-BASS			15. NUMBER OF PAGES 46	
			16. PRICE CODE	
17. SECURITY CLASSIFICATION OF REPORT UNCLASSIFIED	18. SECURITY CLASSIFICATION OF THIS PAGE UNCLASSIFIED	19. SECURITY CLASSIFICATION OF ABSTRACT UNCLASSIFIED	20. LIMITATION OF ABSTRACT UL	

INTENTIONALLY LEFT BLANK.

USER EVALUATION SHEET/CHANGE OF ADDRESS

This Laboratory undertakes a continuing effort to improve the quality of the reports it publishes. Your comments/answers to the items/questions below will aid us in our efforts.

1. ARL Report Number/Author ARL-MR-444 (Schallhorn) Date of Report May 1999

2. Date Report Received _____

3. Does this report satisfy a need? (Comment on purpose, related project, or other area of interest for which the report will be used.) _____

4. Specifically, how is the report being used? (Information source, design data, procedure, source of ideas, etc.) _____

5. Has the information in this report led to any quantitative savings as far as man-hours or dollars saved, operating costs avoided, or efficiencies achieved, etc? If so, please elaborate. _____

6. General Comments. What do you think should be changed to improve future reports? (Indicate changes to organization, technical content, format, etc.) _____

CURRENT ADDRESS	_____
	Organization

	Name E-mail Name

Street or P.O. Box No.	

City, State, Zip Code	

7. If indicating a Change of Address or Address Correction, please provide the Current or Correct address above and the Old or Incorrect address below.

OLD ADDRESS	_____
	Organization

	Name

Street or P.O. Box No.	

City, State, Zip Code	

(Remove this sheet, fold as indicated, tape closed, and mail.)
(DO NOT STAPLE)