TECHNIQUES FOR THE COMPARATIVE ANALYSIS OF DATA FLOW DIAGRAMS

by

Donald J. Berndt
Information Systems Department
Stern School of Business
New York University

February 24, 1993

Center for Research on Information Systems
Information Systems Department
Stern School of Business
New York University

Working Paper Series

STERN IS-93-6

Techniques for the Comparative Analysis of Data Flow Diagrams

D. J. Berndt
Information Systems Department
Stern School of Business
New York University

February 24, 1993

Abstract

This paper presents an analytic framework for comparing data flow diagrams based on five dimensions: control points, process automation, data aggregation, resource usage, and raw counts. Our goal was to develop some simple quantitative metrics that are appropriate for computer-aided system development tools. In addition, we argue for computer-aided tools that support the tandem development of alternative system diagrams. Simultaneous development of competing system descriptions may allow for more accurate contrasts and insightful analysis. Finally, we use two case studies to illustrate the comparison techniques.

1 Introduction

The data flow diagram (DFD) and other related graphical tools are often used to represent complex systems. In many cases, these tools are utilized by analysts and designers during the development of new systems [LJ92] [You89]. The implementation of new organizational systems is usually done in the context of some existing and on-going system environment. That is, we are often replacing or re-engineering a process [DS90] [Ham90]. How are we to compare our existing systems with those that we propose? This question is important during the design stage as well as the post-implementation evaluation. While the tools, such as DFDs, have become somewhat standardized, there is a lack of standard measures for comparing alternative systems. In this paper, we develop some techniques for explicitly comparing alternative system DFDs.

The metrics developed rely on careful consideration by the analyst, just as in the construction of DFDs. Indeed, there is no way to quantify or automate the subjective assessment of what a function or DFD element represents. However, if the DFDs are developed with an eye toward consistency of meaning, the metrics we propose are easily quantifiable and their calculations can be automated.

1.1 Tandem DFD Development

In order to assist the analyst in developing internally consistent descriptions of original and proposed systems, we propose allowing the descriptions to be developed in tandem. That is, our automated tools should support such an approach.

The tools should allow a "core" DFD (representing system overlap) to be constructed, with the alternative system DFDs encoded as points of departure. Simply being able to label the entities and relationships of existing diagraming tools would help in supporting tandem development. The tool could then be used to select "views" by specifying which alternative system(s) you intend to view. The increased diagram complexity could be hidden through proper tool construction.

2 DFD Comparative Analysis

Assuming we wish to compare two systems, what are some meaningful metrics that might serve to guide our analysis? We propose five broad dimensions for comparisons, as well as more precise metrics within these dimensions that can be collected (often through automated techniques).

- *control points*
- *process automation*
- *data aggregation*
- *resource usage*
- *raw counts*

2.1 Control Points

The ability to control our systems is central to successful implementation and affects the level of quality achievable in the outputs. How can we quantify the "grain size" of our control? The notion of *control roots* and *control points* may offer a reasonable method. Counts of both control roots and control points may serve as meaningful metrics. We assume the following operational definitions for our analysis.

- *control root*—An object that *produces* information that is utilized at later points for verification purposes.
- *control point*—An object that *consumes* information and performs a verification function.

We can organize our functions by control root or point, and by level of automation. At first, we only considered manual and automated categories, but introducing a computer-aided category allowed a more refined analysis. Manual and computer-aided categories both require human presence, but computer assistance should enhance performance. Fully automated functions can be accomplished without constant guidance by a human. Further analysis of process automation is the subject of Section 2.2. The matrix presented in this section can be considered a special case of the SON matrix that follows.[1]

[1] We envision our analytic framework as part of a computer-based DFD development environment. Views over the SON matrix could be filtered to display control roots or control points.

automation category / *control category*	manual controls	computer-aided controls	automated controls	totals
old system				
control *roots*	r_1	r_2	r_3	r_t
control *points*	p_1	p_2	p_3	p_t
new system				
control *roots*	r'_1	r'_2	r'_3	r'_t
control *points*	p'_1	p'_2	p'_3	p'_t

Table 1: control root/point matrix.

One possible numeric comparison of *control capability* is simply the old and new totals—$(r_t+p_t) < (r'_t+p'_t)$.[2] Another important metric would be to assess *control automation.* One simple comparison would be $(r_2 + r_3)/r_t < (r'_2 + r'_3)/r'_t$ (and similarly for control points).

Of course, quantitative comparisons based on diverse collections of elements, such as control points, can be risky (or meaningless). Our framework, as well as other system development tools, depend on the subjective description of processes as systems are analyzed. In order to make the comparison more meaningful, the analyst may wish to attach levels of importance to the various objects. The idea of weighting objects is relevant in the next section as well, and is discussed in Section 2.6.

2.2 Process Automation—SON Analysis

The level of automation achieved in our systems is an important measure of effective technology use. In addition, we are interested in the relative change or disruption that alternative systems represent. In order to quantify these concepts, we develop the notions of *stable*, *obsolete*, and *new* functions. Stable functions exist in both the old and new systems, though these functions may shift among automation categories. These functions have been "paired" with their counterparts in the opposing system. Obsolete functions exist in the old system, but have been removed from the new system. New functions are those that appear solely in the new system—hopefully representing

[2]We assume an old (existing) system and a new (proposed) system in our examples, which is often the case in real applications. However, our framework is intended for comparison of any pair of proposed systems.

enhancements. The *SON matrix* (see Figure 2) is a framework for arriving at numerical estimates for these constructs. The analysis is founded on our ability to "pair" the objects from alternative DFDs. The ability to develop DFDs simultaneously, with shared "core" regions, lends itself to this analysis.[3]

automation category / *function category*	manual functions	computer-aided functions	automated functions	totals
stable functions				
old system	s_1	s_2	s_3	s_t
new system	s_1'	s_2'	s_3'	
obsolete functions	o_1	o_2	o_3	o_t
new functions	n_1	n_2	n_3	c_t

Table 2: SON matrix.

The *stability* factor is intended to capture the extent to which two alternative systems overlap. This type of measure can be thought of as a measure of the "radicalness". The measure is calculated as $s_t/(s_t + o_t + n_t)$. We can derive a measure of *obsolescence* (i.e. retired functions) as $o_t/(s_t + o_t + n_t)$. Lastly, $n_t/(s_t + o_t + n_t)$ is a measure of *newness*. We might loosely think of it as innovativeness, but this seems a bit presumptuous.

2.2.1 Level of Automation

In addition to simple ratios described above, we can use the change in automation categories within the stable functions as one measure of *process automation*. We would expect a new system to employ more automation, so $(s_2 + s_3)/s_t < (s_2' + s_3')/s_t$. In this calculation, we consider both computer-aided and automated functions as affecting the extent of automation.

2.2.2 System-wide Change

Lastly, our earlier measure of newness is based only on new functions—possibly understating the overall change between two alternative systems. In order to quantify system-wide change, we would like to combine counts of new (as well as obsolete) functions with the extent of automation, as

[3]In the absence of such support, the analyst must develop a pairing manually.

reflected in the shifting of stable functions among automation categories. To capture the amount of change within stable functions, we propose the following: $\sum \mid s_i - s_i' \mid /2$. This summation captures the change between the old and new systems in each automation category (halved to correct for double counting). We can then form a new measure of overall or *system-wide change* using the following equation.[4]

$$\frac{\sum \mid s_i - s_i' \mid /2 + o_t + n_t}{s_t + o_t + n_t} \tag{1}$$

2.3 Data Aggregation

This dimension is concerned with the specificity or abstraction that is represented in the DFD data stores, as well as a similar concern for automation. As we design new systems, we are often interested in developing a "finer" level of data detail. Computer technology obviously supports rather dramatic changes in the volume of data that can be manipulated. We hope to develop some type of ranking that will allow a quantitative assessment of the change in available data aggregation.

Currently, we simply categorize data stores as stable, obsolete, or new. We replace the earlier automation categories with the form of storage technology used: *physical* or *electronic*. That is, stable data stores may shift between physical and electronic form in the old and new systems. Physical data stores are used to hold everything from paper-based information to actual inventory. Electronic data stores hold information in a form that can be easily manipulated and transmitted. In addition, each data store has a comment indicating the level of data *aggregation* available for manipulation. This approach allows new and higher-resolution data stores to be highlighted in the analysis. This information can then be encoded in a SON-type matrix with the ratios calculated as before.

2.4 Resource Usage

We would also like to incorporate more traditional measures into our comparative framework, such as staffing levels or process duration. These measures are found in operations management aids such as PERT/CPM and Gantt charts [MT85]. Time and labor are two useful measures of the effort involved in accomplishing a given task. We propose attaching values for

[4]By considering the simple ratios for obsolesence and newness, the analyst can identify the largest contributors to the system-wide change metric.

labor and time to terminal processes in our DFDs. It should be useful to include the maximum, minimum, and average labor and time required. The values can then be propagated throughout the tree of diagrams, so the analyst can view the aggregate values associated with higher-level processes. The analyst can then compare labor and time requirements for chains of processes in alternative systems.

2.5 Raw Counts

An easily collected, but perhaps less meaningful group of metrics, include raw counts of the DFD elements. This process can be automated, while the responsibility for interpretation is left with the analyst.

2.6 Weighting Functions

When constructing a DFD, the analyst tries to be consistent in the level of function detail depicted at each diagram level. However, the analyst may feel certain functions are relatively more important. A potential for systematic bias exists due to the fact that we equally weight all DFD functions. To correct for this, it seems simply a matter of introducing a weighting that more accurately reflects our assessment. How should we develop a series of weights? This process can quickly become unmanageable and often relies on subjective estimates. We propose developing our analysis on a level-by-level basis in the DFD tree structure and allowing weights to be attached or computed for each function. The simplest method is to allow user developed estimates to be attached to functions (with support for some automation akin to spreadsheet calculations). A second repercussion is that all of the metrics discussed above should be available in both aggregated or level-by-level form.

We also propose a second approach that relies on a heuristic rule requiring tree traversal algorithms. In general, DFDs are developed with the more complex or "important" functions expanded, which represents lower-level subtrees in our DFDs. A simple heuristic to determine the relative "weight" of a function is to recursively descend the attached subtree and count the nodes (or terminal nodes) discovered. We could use this count directly, or perhaps adjust it by dividing the number of nodes discovered by the total number of nodes (or terminal nodes at each level). Such numbers may provide rough estimates of function importance.

3 Case Study—Accounts Payable System

Hammer [Ham90] presents an important discussion of re-engineering. One of his illustrations is based on an analysis of Ford's accounts payable system. The high-level DFDs used by Hammer have been adapted to serve as a simple case study for our metrics. (see Appendix A). The systems are depicted at a very high level, rendering the metrics equally abstract. In practice, we would expect a much more thorough set of diagrams in an on-going re-engineering effort (see Section 4). The SON matrix for these simple diagrams is shown in Table 3.

Ford re-engineered the accounts payable system by drastically lowering the paper-based communication within the system. The major functions remained the same—how they did business was radically changed. The major enhancement was the introduction of an electronic database of purchasing information, accessible to all related departments. Since the major functions were constant, our metrics indicate a highly stable core (see Table 4). However, the measures of process automation and system-wide change reflect the new electronic methods of handling accounts payable.

automation category / *function category*	manual functions	computer-aided functions	automated functions	totals
stable functions				
old system	4	0	0	4
new system	1	3	0	
obsolete functions	0	0	0	0
new functions	0	0	1	1

Table 3: Ford accounts payable system SON matrix.

With regard to control points, the situation is similar (see Table 5). The major control functions are retained, but benefit from automation via the electronic database of purchasing information. The details of these functions can only be imagined given the high-level descriptions, however, the case does provide a useful introduction to the metrics.

stability	$4/5 = 0.8$
obsolesence	$0/5 = 0.0$
newness	$1/5 = 0.2$
process automation	$0/4 = 0.0 < 3/4 = 0.75$
system-wide change	$4/5 = 0.8$

Table 4: summary SON metrics for accounts payable system.

automation category	manual	computer-aided	automated	
control category	controls	controls	controls	totals
old system				
control *roots*	2	0	0	2
control *points*	1	0	0	1
new system				
control *roots*	0	2	0	2
control *points*	0	1	0	1

Table 5: Ford accounts payable system control matrix.

4 Case Study—Securities Processing

Merrill Lynch is among the largest depositories for securities such as stock certificates and bonds—handling some 3500 securities each day from over 400 branches. The securities must be carefully handled since some are negotiable by the bearer and all are monitored by the Securities and Exchange Commission (SEC). The goal is to credit the securities to the customer's account as quickly as possible, certainly within the 24 hour deadline specified by the SEC. Merrill Lynch has recently developed a new securities processing system that involves digital imaging technology. Digitized images of the securities (and supporting documents) are archived on optical disk for easy retrieval and transmission, while the actual securities stay in the vault. Character recognition on several fields of the image is used to provide an automated check on the flow of securities. This system is intended to remove the need for the physical securities to be moved, where they are more susceptible to loss or theft, and require a detailed audit trail (including microfilm). We have developed DFDs for the old and new systems that can be analyzed along the dimensions discussed above (see Appendix B).

The top-level DFD functions show one of the largest differences between the two systems. The geographical consolidation of securities processing into a single site in New York, dismantling the Securities Processing Centers in Chicago and Philadelphia. At this level, it makes no sense to move beyond the obvious qualitative comparison. We begin most of the quantitative comparison at the next level, matching the higher-level functions. The following sections describe the process.

4.1 Control Points

The imaging system, the major technological enhancement, provides additional control roots/points. One new computer-aided control root is provided by the on-line document collection system employed at the branch offices. This function used to be handled with typed (carbon-copied) receipts. Secondly, a legal expert system assists branch employees in collecting the proper documents, providing an important new control function. Finally, a new control root is the optical store of digitized securities images. Character recognition on these digitized images provides a new, and quite detailed control point. In fact, the on-line images of securities (and supporting documents) give Merrill Lynch efficient access to certificate-level information.

The numeric comparisons are presented in Table 7. Numerically, the

automation category	manual	computer-aided	automated	
control category	controls	controls	controls	totals
old system				
control *roots*	2	0	0	2
control *points*	2	0	1	3
new system				
control *roots*	0	1	3	4
control *points*	1	1	3	5

Table 6: Merrill Lynch securities processing system control matrix.

new system exceeds the old system in both control roots and control points. More importantly, computerization of control functions has improved.

control root capability	$2 < 4$
control point capability	$3 < 5$
control root automation	$0/2 = 0.0 < 4/4 = 1.0$
control point automation	$1/3 = 0.33 < 4/5 = 0.8$

Table 7: summary control metrics for securities processing system.

4.2 SON Analysis

We used the two DFDs developed for Merrill Lynch and categorized the functions for use in the SON matrix. The counts are shown in Table 8. The summary measures based on these counts are presented in Table 9.

The functions in this first analysis are not weighted by any measure of importance. As discussed in Section 2.6, weighting functions may be an important aspect of the analysis. In this particular case, one could argue that little functionality became obsolete—but whole new functions were added.

4.2.1 Weighting New Functions

New functionality is represented in DFD branches that are not matched by anything in the description of the old system. Weighting these functions

automation category	manual	computer-aided	automated	
function category	functions	functions	functions	totals
stable functions				
old system	10	2	3	15
new system	3	7	5	
obsolete functions	3	0	0	3
new functions	0	0	4	4

Table 8: Merrill Lynch securities processing system SON matrix.

stability	$15/22 = 0.68$
obsolesence	$3/22 = 0.14$
newness	$4/22 = 0.18$
process automation	$5/15 = 0.33 < 12/15 = 0.8$
system-wide change	$14/22 = 0.64$

Table 9: summary SON metrics for securities processing system.

may be of particular importance in assessing obsolesence and newness.[5] We would like to develop automated weighting algorithms that can reflect the importance of new (and obsolete) functions. For example, the new functions that include image capture and character recognition can be weighted by counting the corresponding lower-level functions. Therefore, both functions would have a count of 3 instead of 1—yielding the new summary metrics in Table 10.

stability	$15/26 = 0.58$
obsolesence	$3/26 = 0.12$
newness	$8/26 = 0.31$
process automation	$5/15 = 0.33 < 12/15 = 0.8$
system-wide change	$15/26 = 0.58$

Table 10: summary SON metrics for securities processing system.

5 Conclusions

The analysis of the two case studies show the metrics to be useful and amenable to computer-aided design tools. A second goal was to argue for the construction of design tools that will support the simultaneous development of competing solutions, allowing more precise contrasts. The metrics appear to capture some sense of change with regard to functionality and automation. The simple ratios for stability, obsolescence, and newness highlight differences between the functionality of alternative systems. Metrics for process automation and system-wide change indicate the role of automation in the systems. In the Ford case, the change was predominately through automation of existing functions. Merrill Lynch, on the other hand, was a project that encompassed changes in the functions performed, as well as process automation via expert system and imaging technologies. Both cases are good examples of re-engineering. In addition, control point metrics explicitly address an important area of business systems. The Merrill Lynch project improved in both the number and automation of control points. Of

[5]Currently, we do not use weights. Further experience with these techniques are required before deciding if weights are helpful.

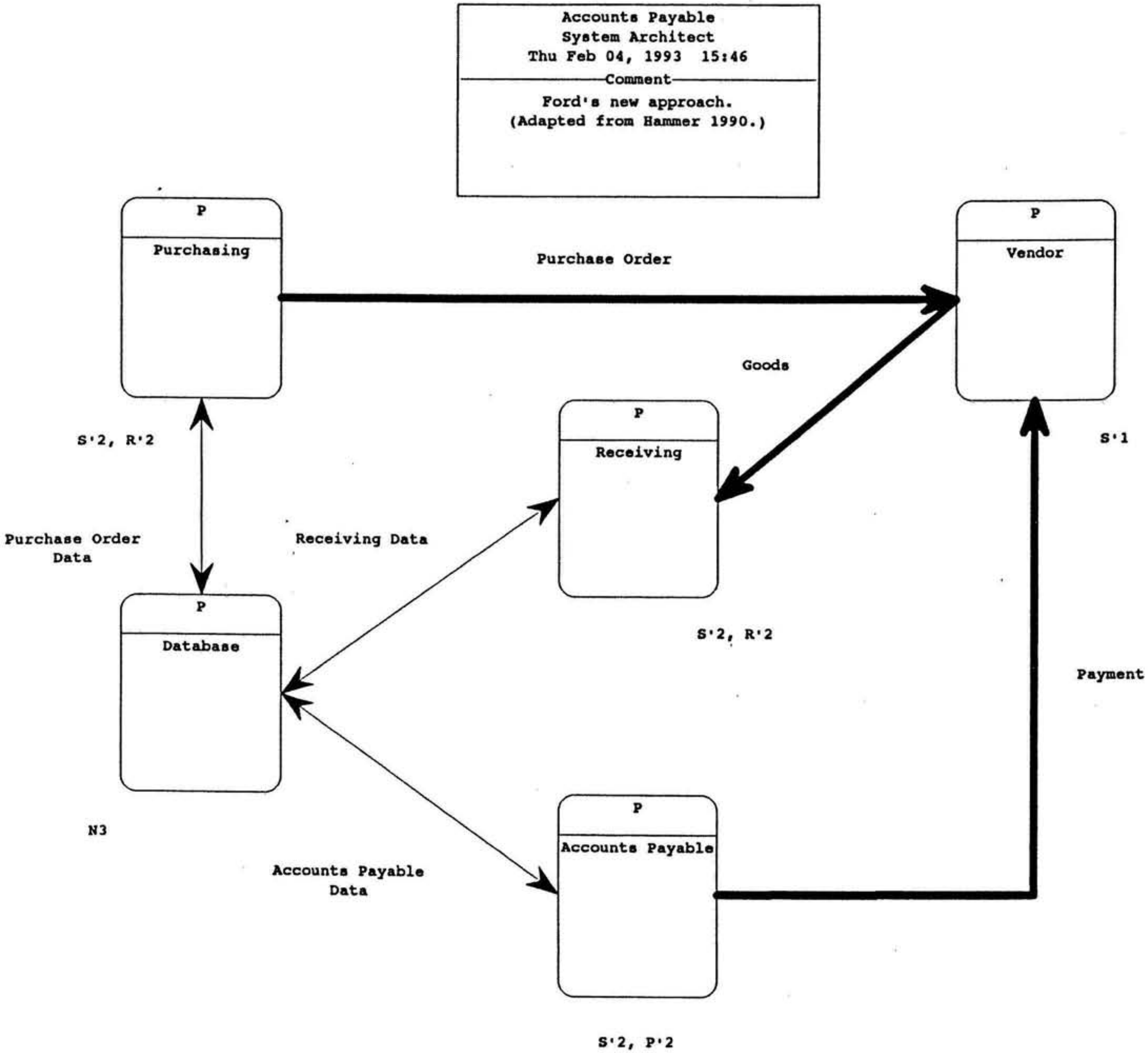
course, the proposed analytic framework will need to be refined and applied to a collection of system development projects before more concrete interpretations of the numeric measures are worthwhile.

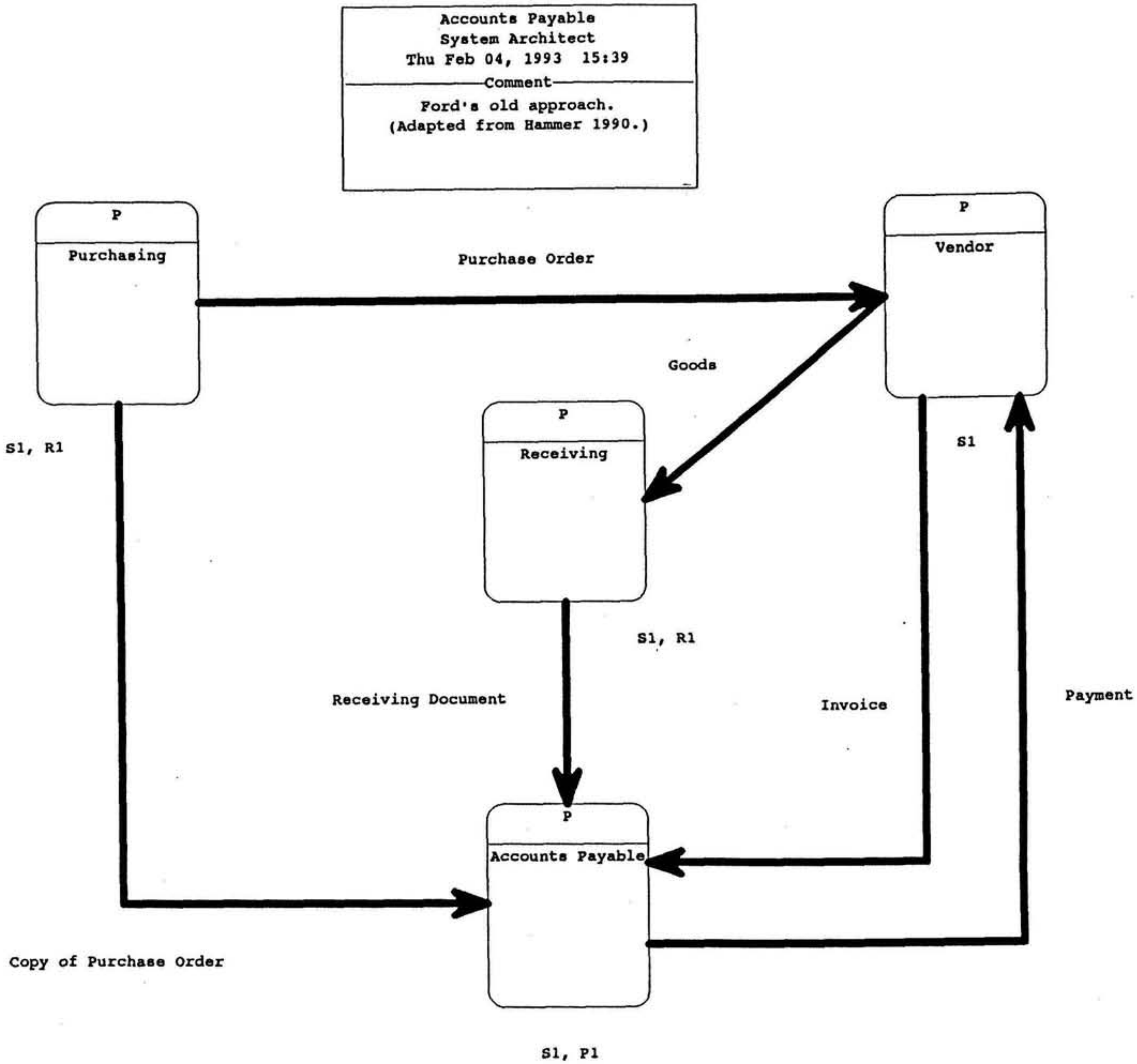
6 Acknowledgements

I would like to thank Hank Lucas and Greg Truman for their advice on this paper and hard work on the Merrill Lynch case study. I am especially indebted to Greg Truman for his partnership in constructing the Merrill Lynch data flow diagrams. Lastly, I would like to thank Safwan Masri and Donald Nagle of Columbia University for their collaboration on the Merrill Lynch case study.

A Ford Data Flow Diagrams

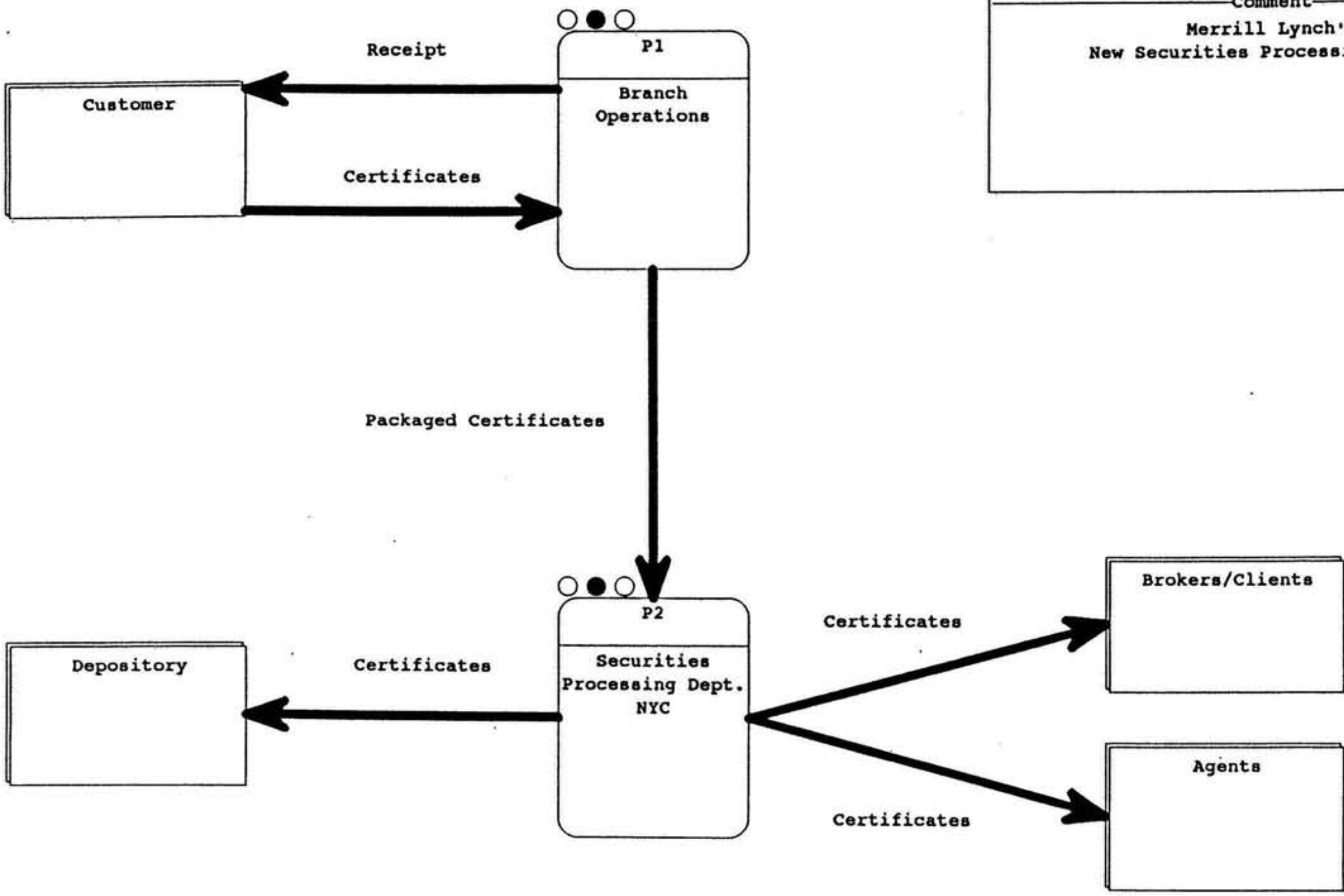
Note: the heavy arrows indicate physical flows and the light arrows indicate information flows. The processes are numbered to reflect their level in the diagram tree, as well as appropriate ancestor processes.


Accounts Payable
System Architect
Thu Feb 04, 1993 15:46
Comment
Ford's new approach.
(Adapted from Hammer 1990.)
P
Purchasing
Purchase Order
P
Vendor
Goods
P
Receiving
S'2, R'2
S'1
Purchase Order Data
Receiving Data
P
Database
S'2, R'2
Payment
N3
P
Accounts Payable
Accounts Payable Data
S'2, P'2


Accounts Payable
System Architect
Thu Feb 04, 1993 15:39
Comment
Ford's old approach.
(Adapted from Hammer 1990.)
P
Purchasing
Purchase Order
P
Vendor
Goods
P
Receiving
S1, R1
S1
S1, R1
Receiving Document
Invoice
Payment
P
Accounts Payable
Copy of Purchase Order
S1, P1

B Merrill Lynch Data Flow Diagrams

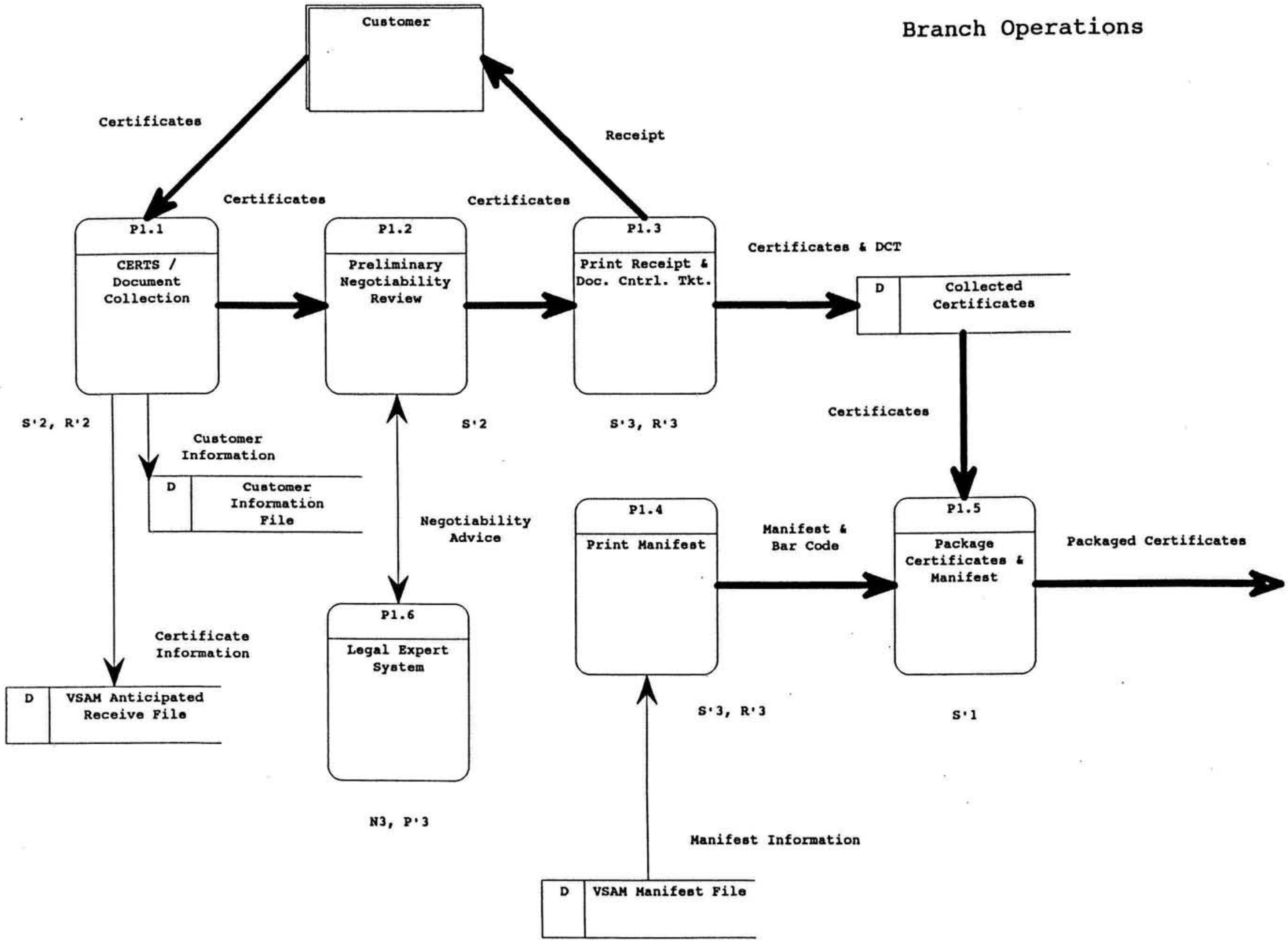
Note: the heavy arrows indicate physical flows and the light arrows indicate information flows. The processes are numbered to reflect their level in the diagram tree, as well as appropriate ancestor processes.



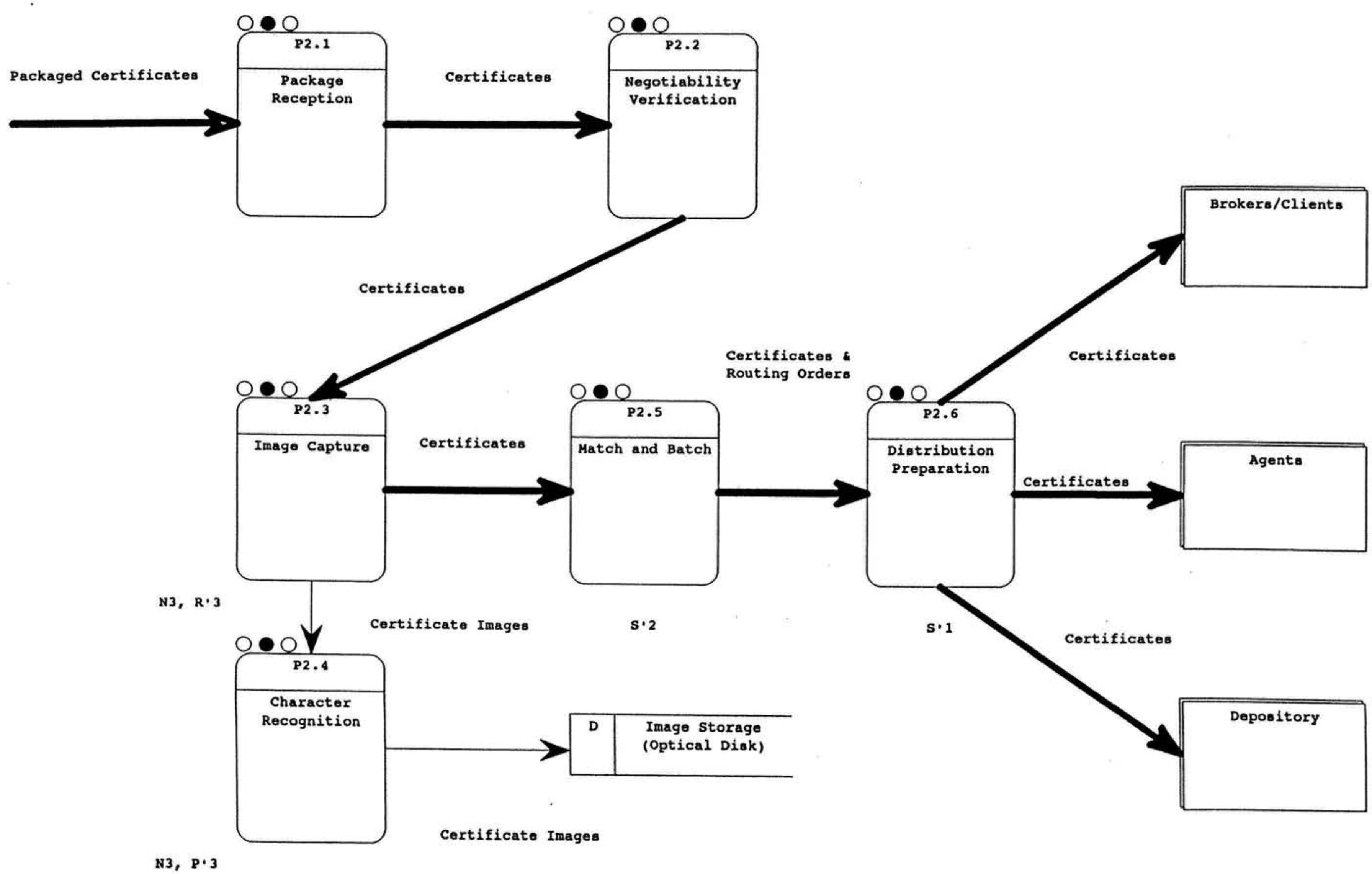
Top Level Diagram
System Architect
Thu Feb 04, 1993 15:51
Comment
Merrill Lynch's
New Securities Processing System

Version 3.5

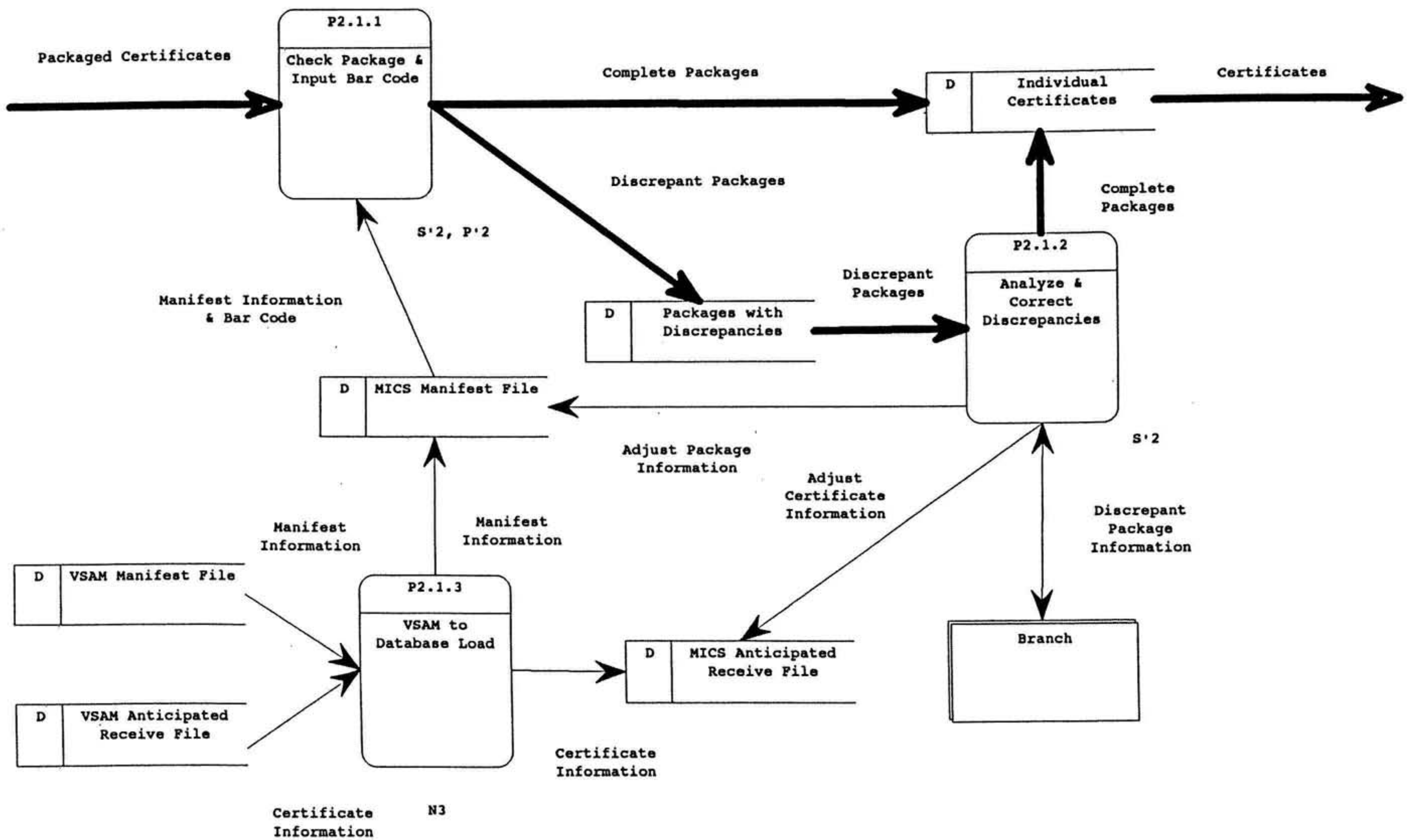
Branch Operations



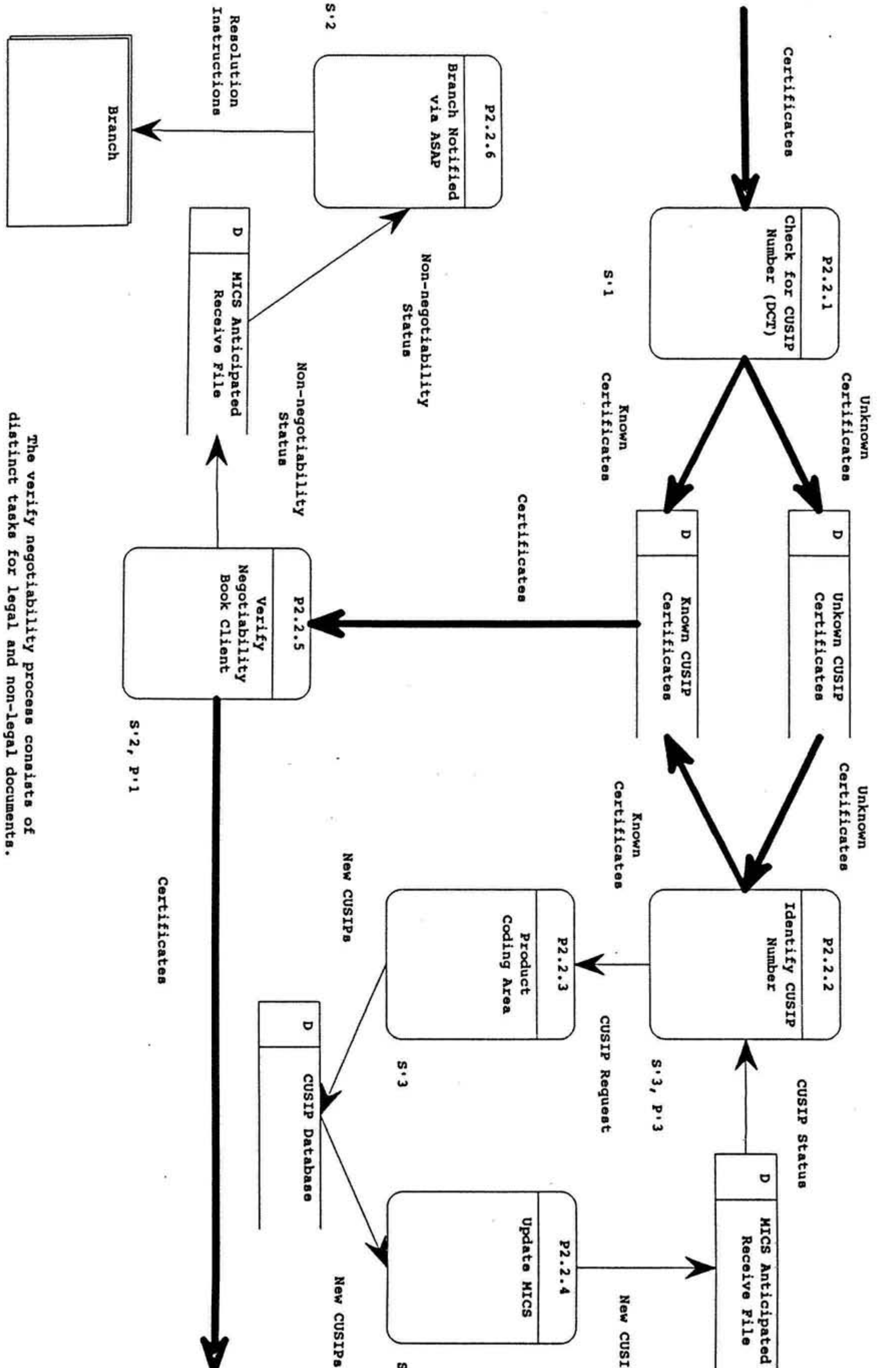
Securities Processing Department



Package Reception

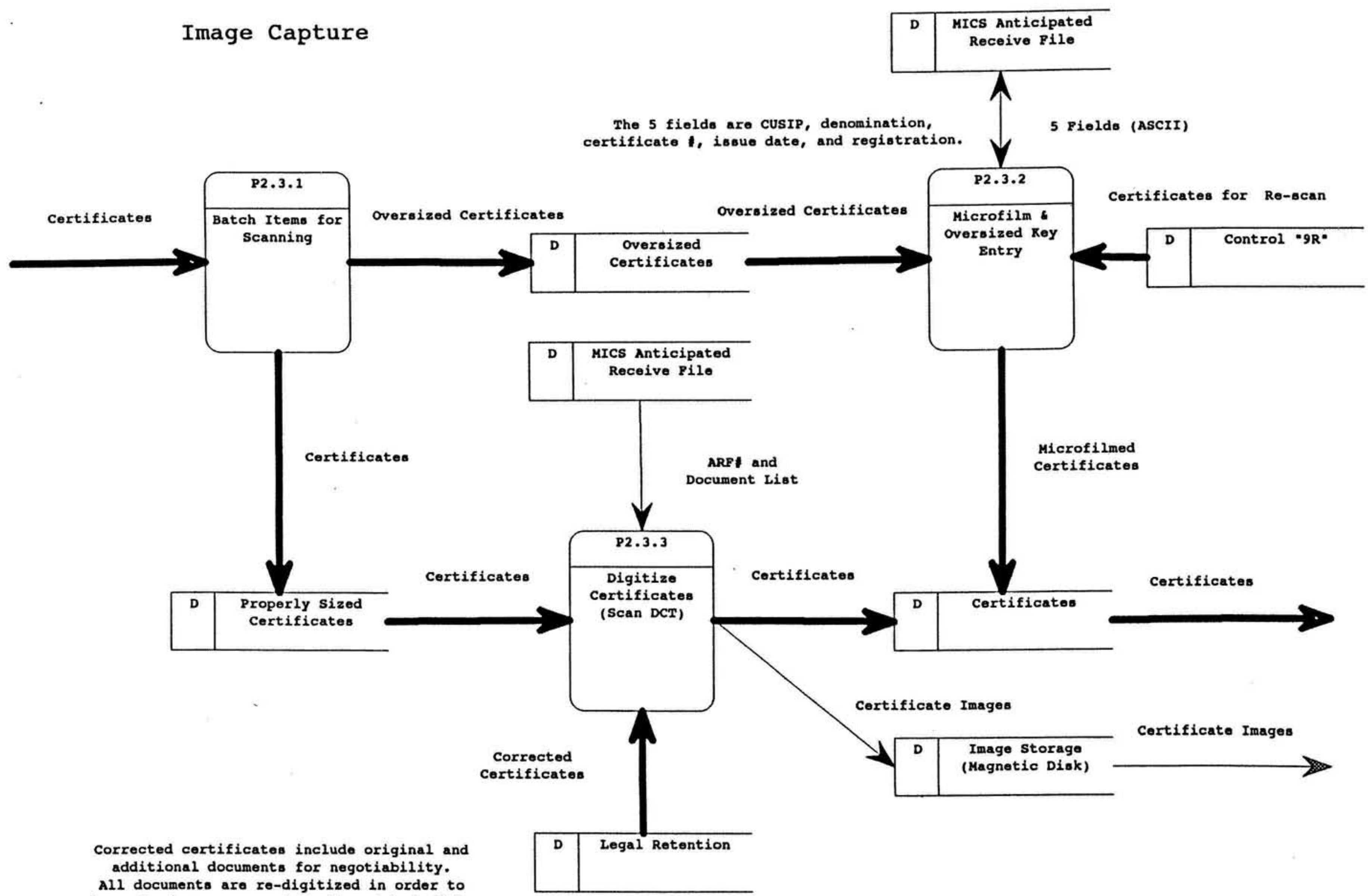


Negotiability Verification



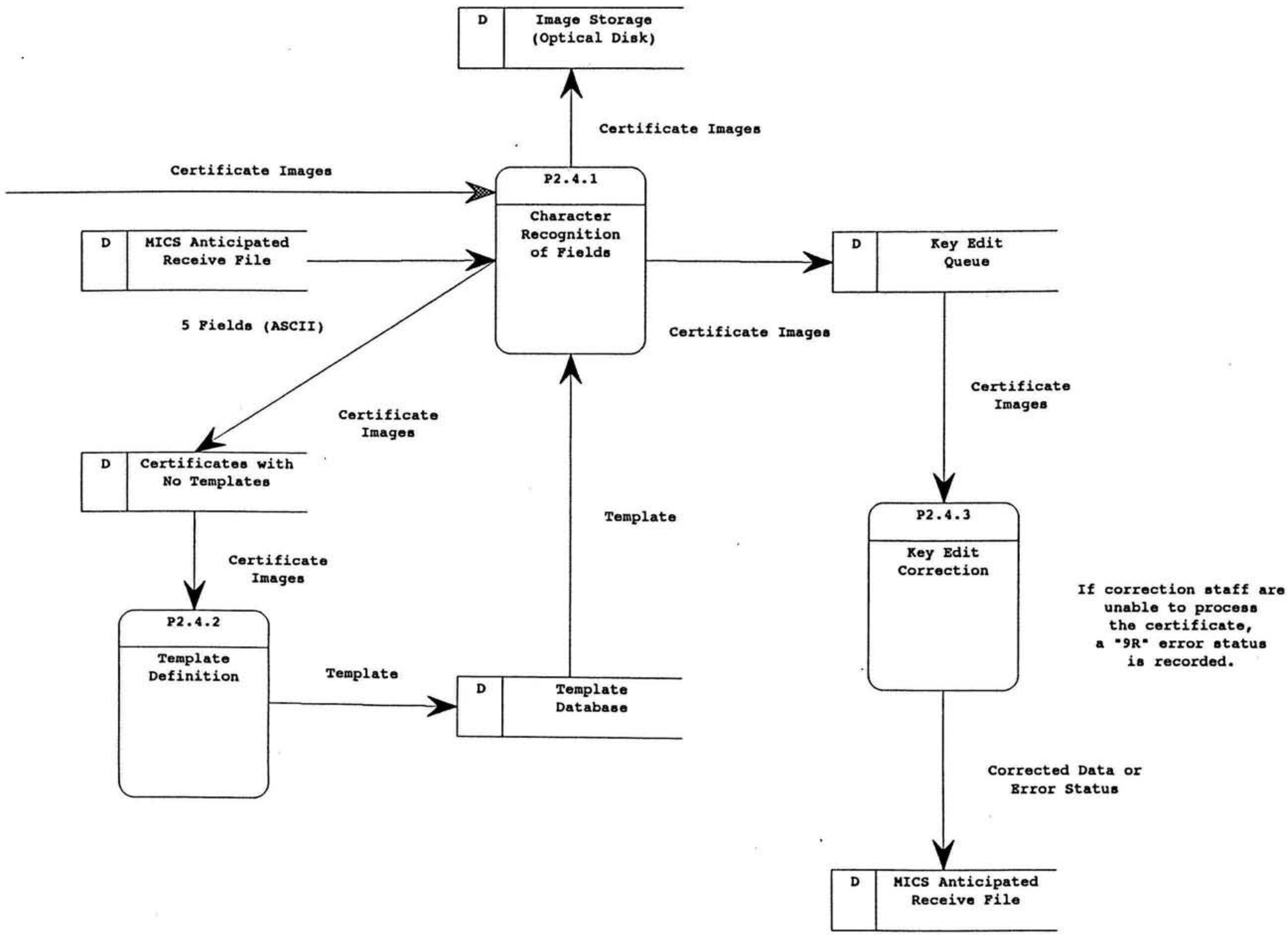
The verify negotiability process consists of distinct tasks for legal and non-legal documents.

Image Capture

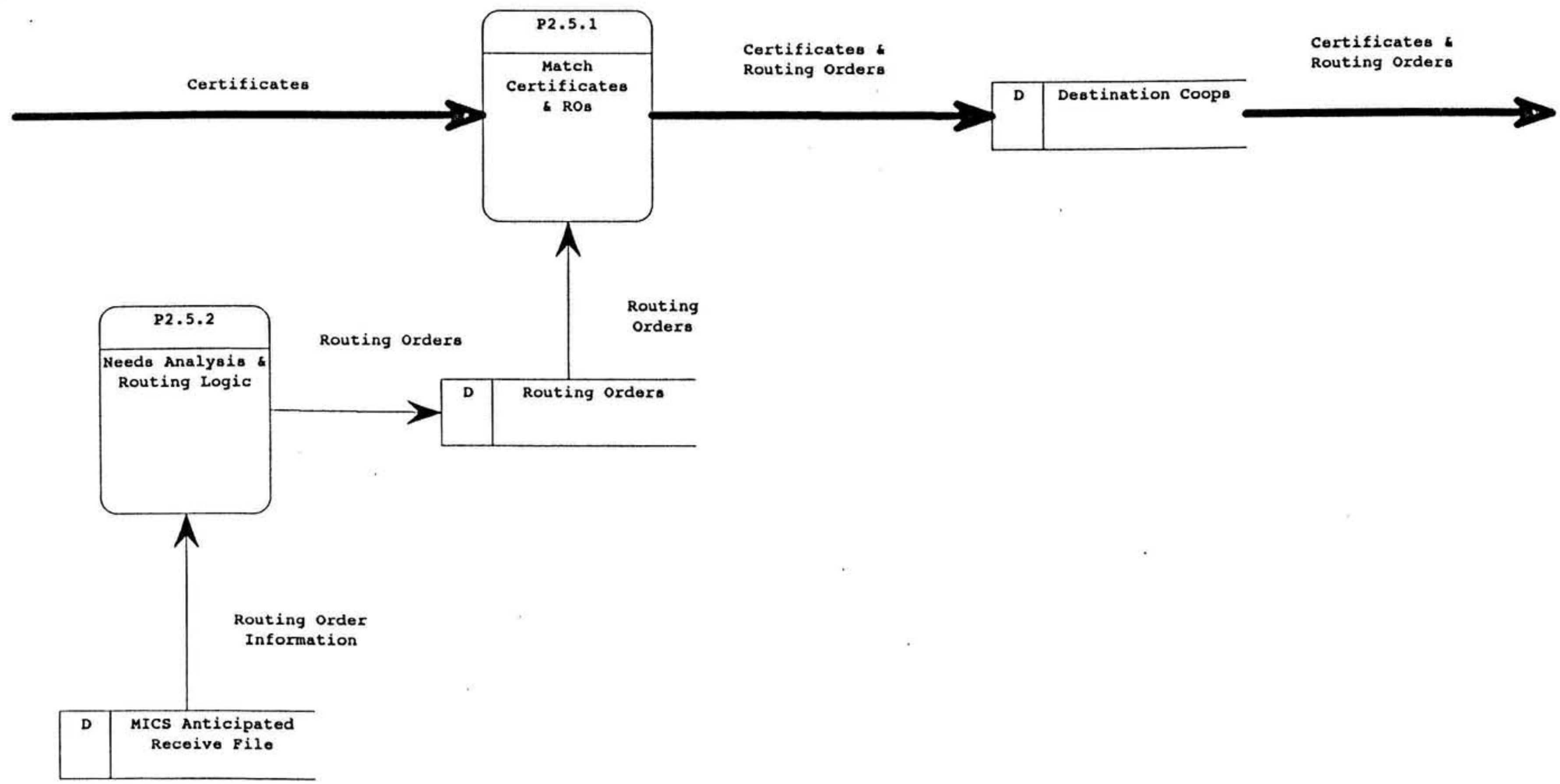


Corrected certificates include original and additional documents for negotiability. All documents are re-digitized in order to keep the images grouped on the optical disk.

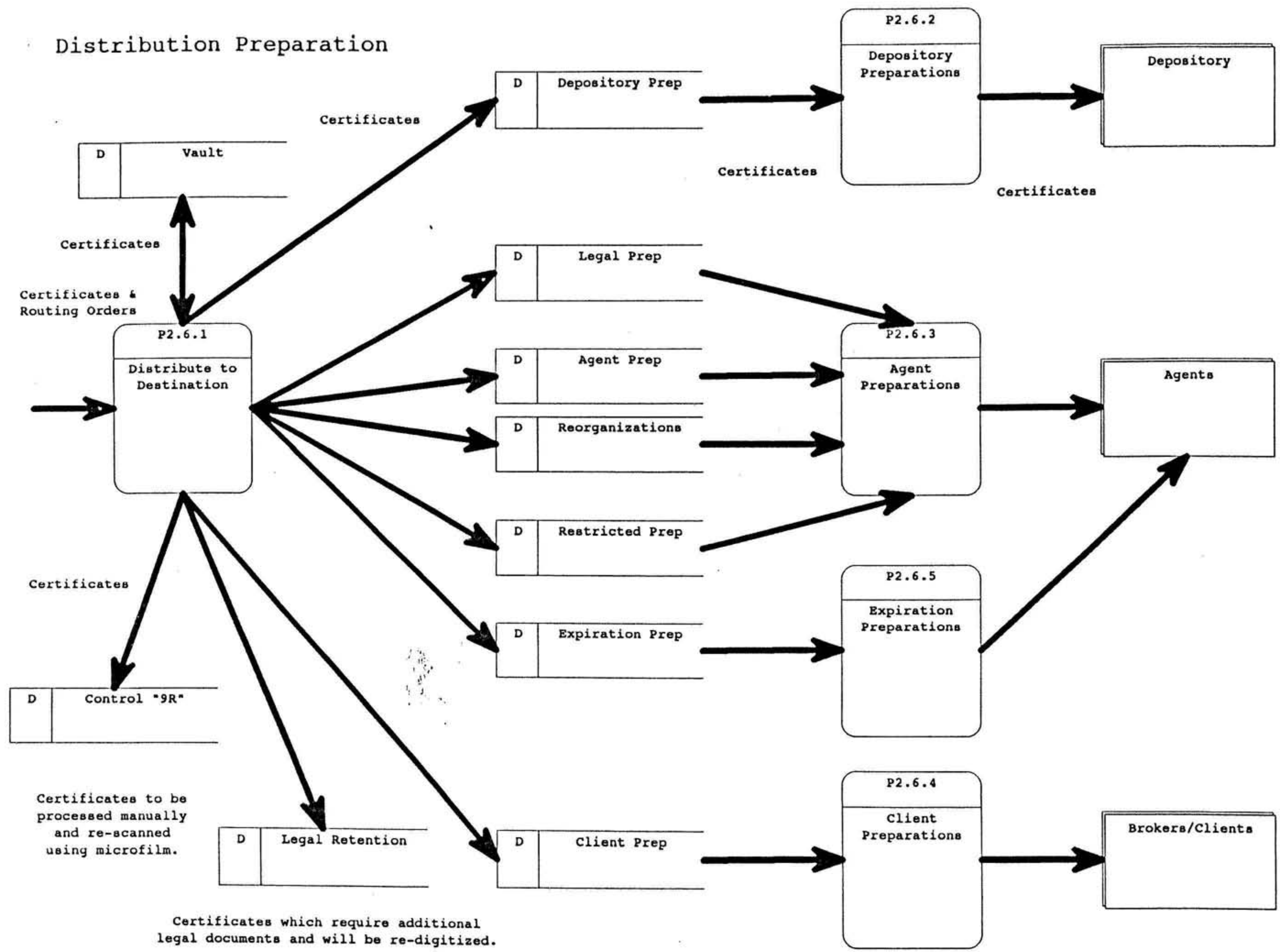
Character Recognition

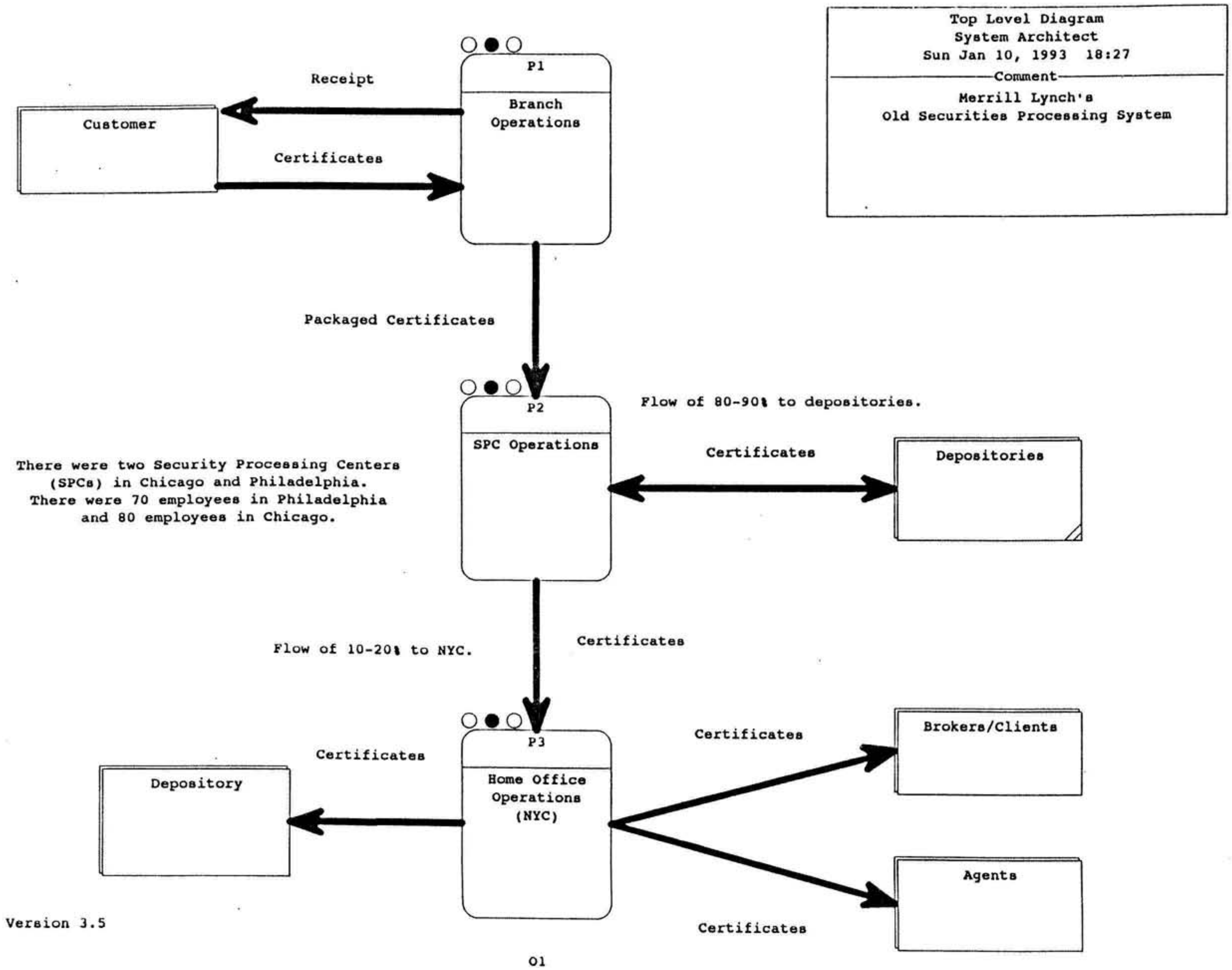


Match and Batch



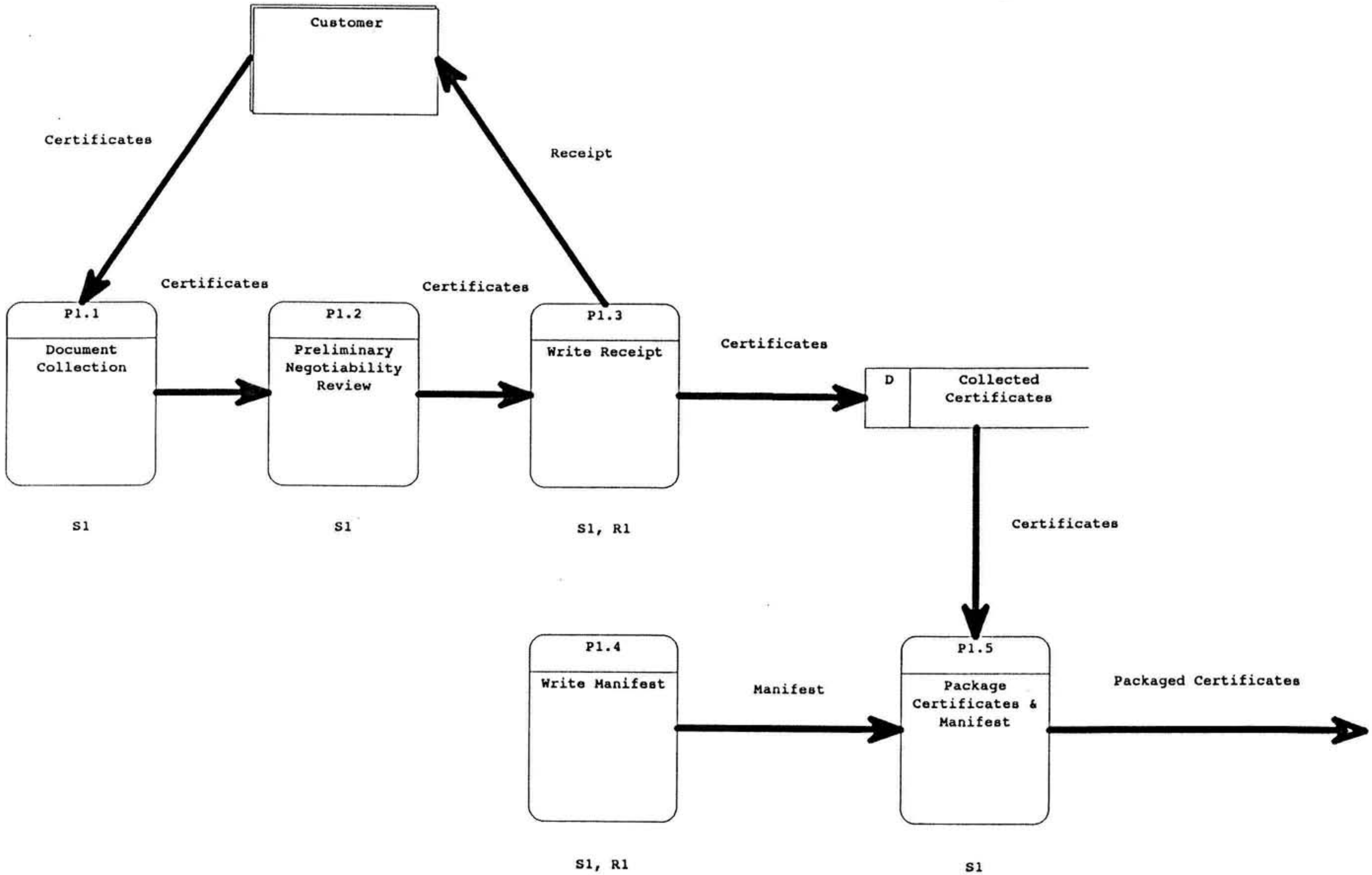
Distribution Preparation



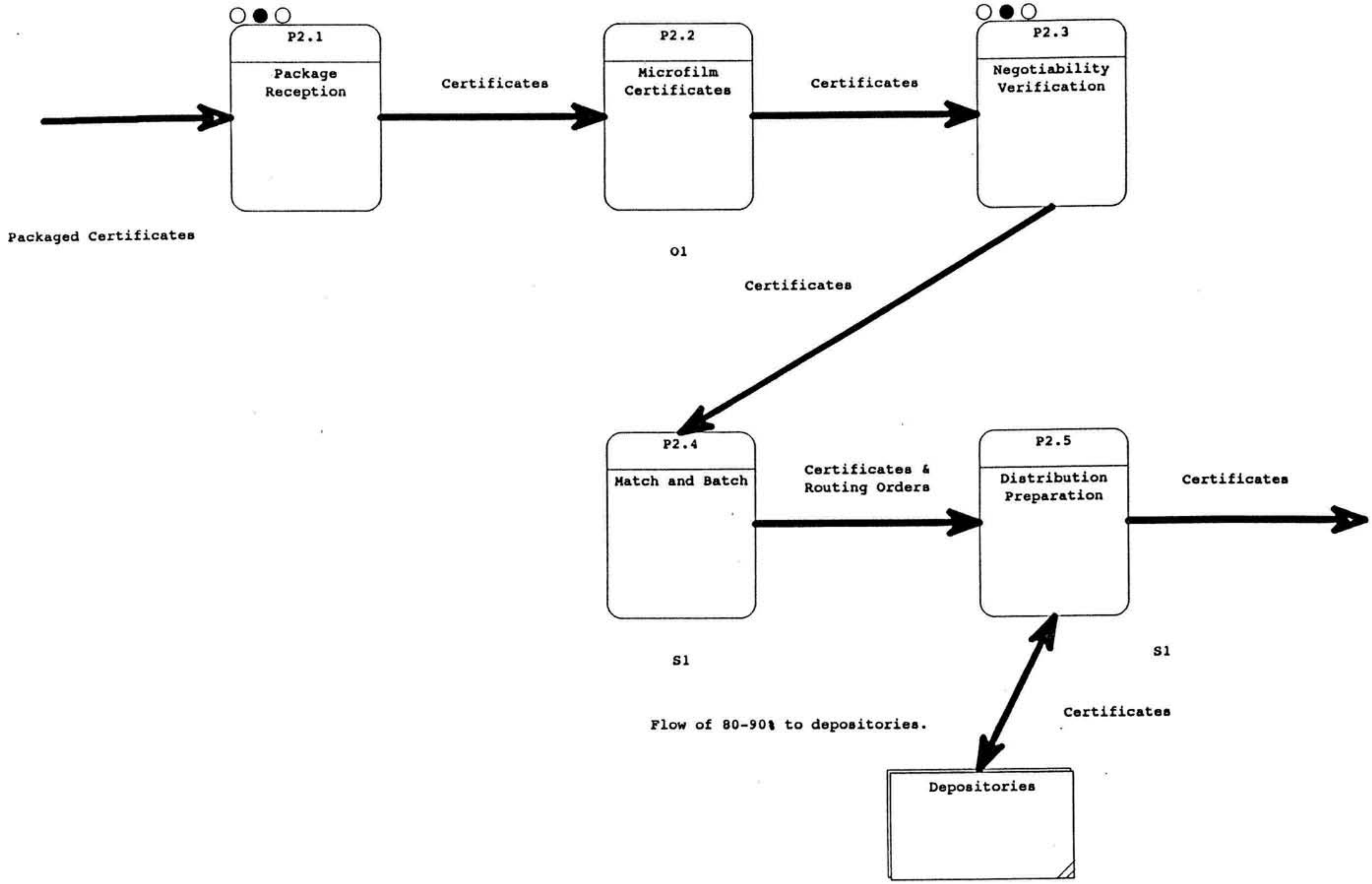


Top Level Diagram
System Architect
Sun Jan 10, 1993 18:27
Comment
Merrill Lynch's
Old Securities Processing System
P1
Branch Operations
Customer
Receipt
Certificates
Packaged Certificates
P2
SPC Operations
Flow of 80-90% to depositories.
Certificates
Depositories
There were two Security Processing Centers (SPCs) in Chicago and Philadelphia.
There were 70 employees in Philadelphia and 80 employees in Chicago.
Flow of 10-20% to NYC.
Certificates
P3
Home Office Operations (NYC)
Depository
Certificates
Certificates
Brokers/Clients
Certificates
Agents
Version 3.5
O1

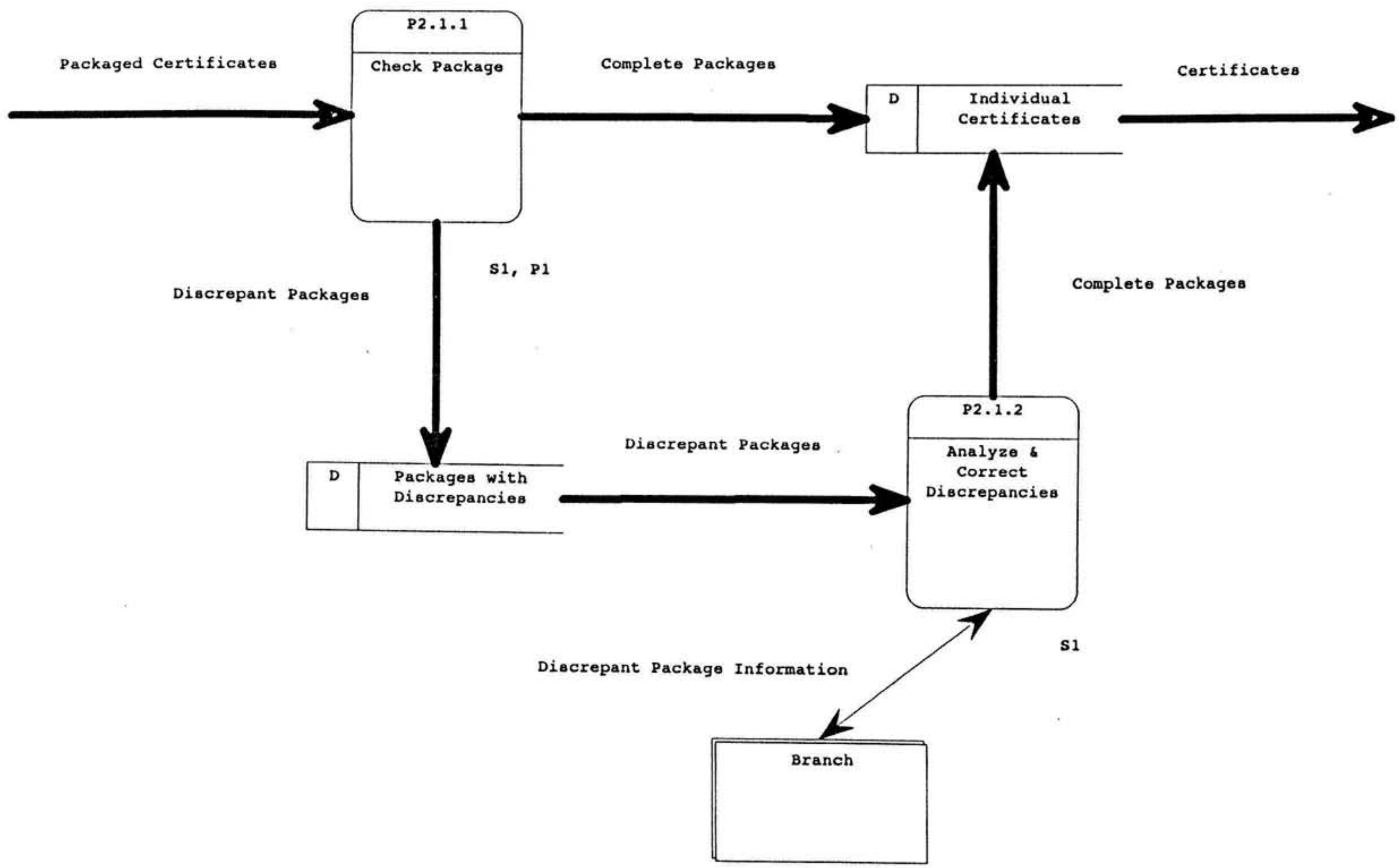
Branch Operations



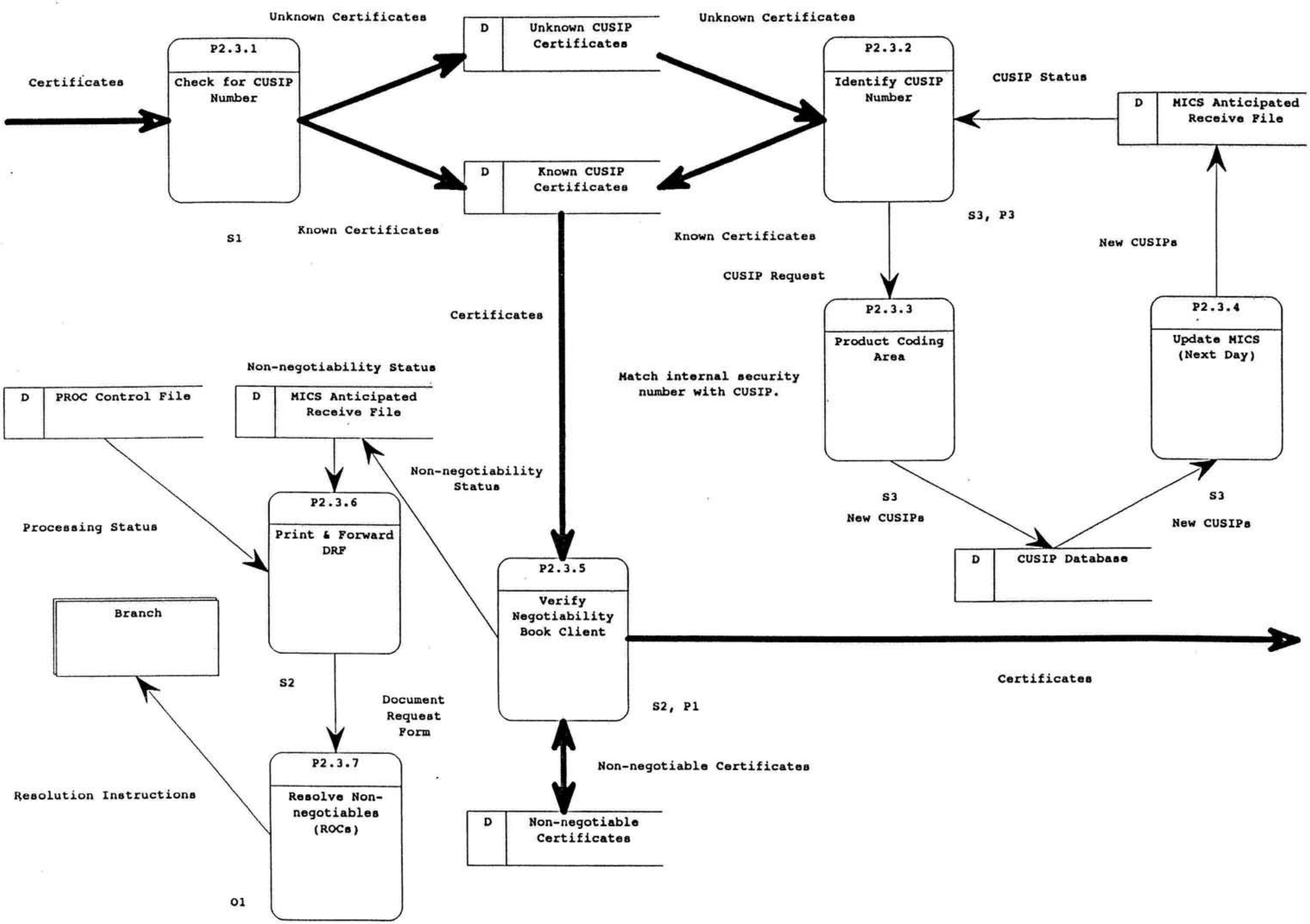
SPC Operations



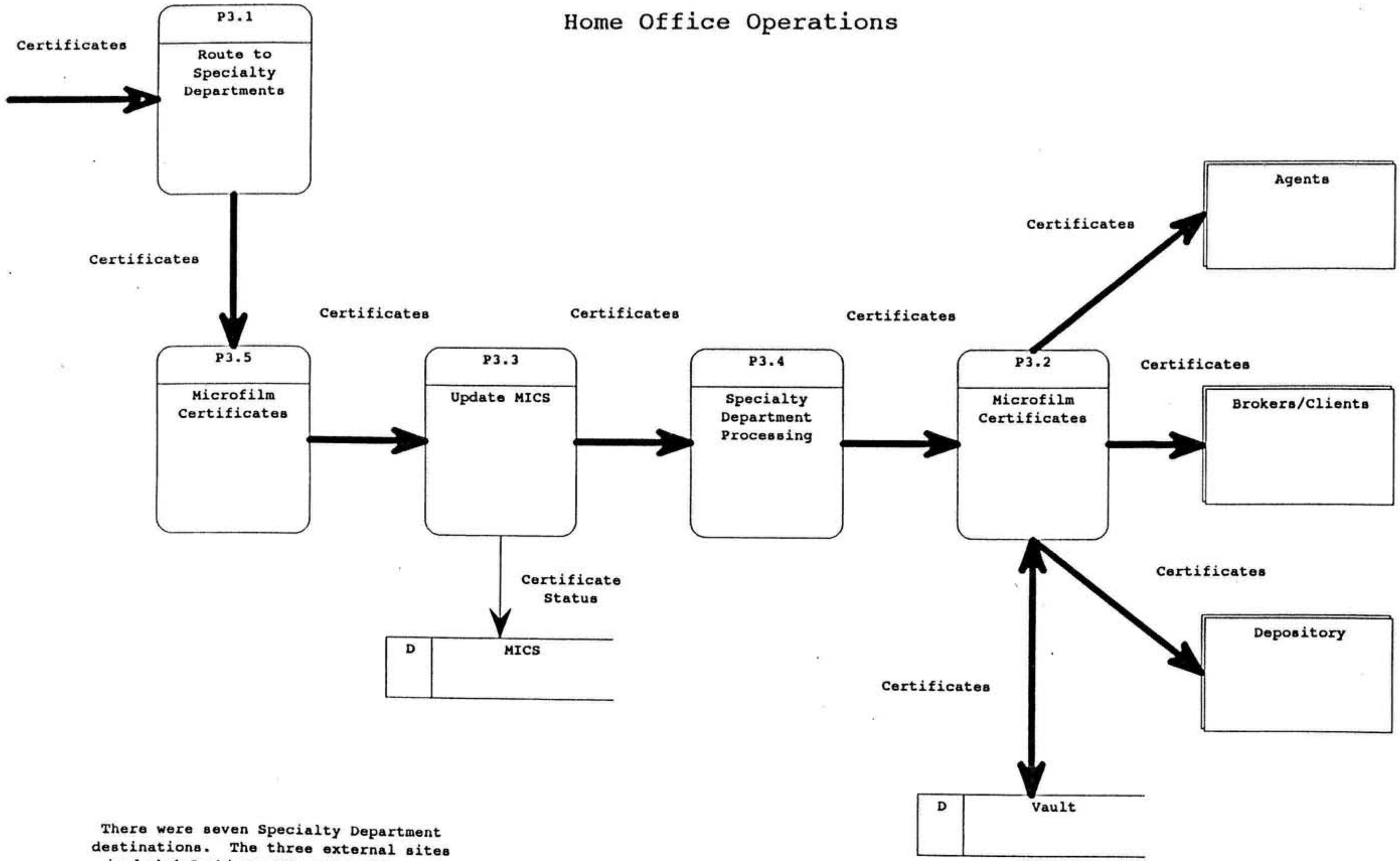
Package Reception



Negotiability Verification



Home Office Operations



There were seven Specialty Department destinations. The three external sites included Cashiers OLP, GMMD OLP, and Somerset NJ. The internal sites included Transfer Services, Legal Transfer, Exchange, and Expiration. Each of these departments microfilmed and processed certificates before routing to final destinations.

References

[DS90] Thomas H. Davenport and James E. Short. The new industrial engineering: Information technology and business process redesign. *Sloan Management Review*, 31(4):11–27, Summer 1990.

[Ham90] Michael Hammer. Reengineering work: Don't automate, obliterate. *Harvard Business Review*, pages 104–112, July–August 1990.

[LJ92] Henry C. Lucas Jr. *The Analysis, Design, and Implementation of Information Systems.* McGraw-Hill, New York, fourth edition, 1992.

[MT85] John O. McClain and L. Joseph Thomas. *Operations Management.* Prentice-Hall, Englewood Cliffs, New Jersey, 1985.

[You89] Edward Yourdon. *Modern Structured Analysis.* Prentice-Hall, Englewood Cliffs, New Jersey, 1989.