

University of Montana

ScholarWorks at University of Montana

Graduate Student Theses, Dissertations, &
Professional Papers

Graduate School

1987

Automated undergraduate advising system

Tsui-Shien Amy Wang

The University of Montana

Follow this and additional works at: <https://scholarworks.umt.edu/etd>

Let us know how access to this document benefits you.

Recommended Citation

Wang, Tsui-Shien Amy, "Automated undergraduate advising system" (1987). *Graduate Student Theses, Dissertations, & Professional Papers*. 8155.

<https://scholarworks.umt.edu/etd/8155>

This Thesis is brought to you for free and open access by the Graduate School at ScholarWorks at University of Montana. It has been accepted for inclusion in Graduate Student Theses, Dissertations, & Professional Papers by an authorized administrator of ScholarWorks at University of Montana. For more information, please contact scholarworks@mso.umt.edu.

COPYRIGHT ACT OF 1976

THIS IS AN UNPUBLISHED MANUSCRIPT IN WHICH COPYRIGHT
SUBSISTS. ANY FURTHER REPRINTING OF ITS CONTENTS MUST BE
APPROVED BY THE AUTHOR.

MANSFIELD LIBRARY
UNIVERSITY OF MONTANA
DATE: 1987

AN AUTOMATED UNDERGRADUATE ADVISING SYSTEM

By

Tsui-Shien Amy Wang

B.A., Tamkang University, 1982

Presented in partial fulfillment of the requirements

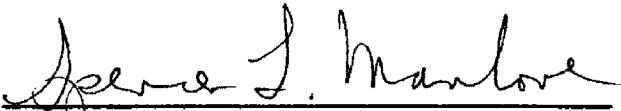
for the degree of

Master of Science

University of Montana

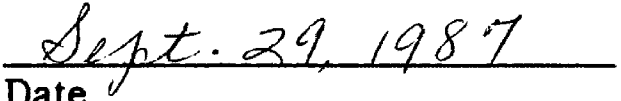
1987

Approved by



Chairman, Board of Examiner


Dean, Graduate School


Date

UMI Number: EP38956

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



UMI EP38956

Published by ProQuest LLC (2013). Copyright in the Dissertation held by the Author.

Microform Edition © ProQuest LLC.

All rights reserved. This work is protected against unauthorized copying under Title 17, United States Code



ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 - 1346

I. Introduction	1
Advising Problems.....	1
Proposed AUAS.....	2
System Goals	3
Graduation Requirements	4
II. System Overview.....	5
AUAS Data Base Structure.....	5
AUAS Functions.....	9
AUAS Program Structure	12
III. Implementation Method	19
The Problem to be Solved.....	19
The Transformation Operation.....	19
Transformed Requirement Data Base Structure.....	25
IV. Examples	28
Understanding more about AUAS data bases	28
Time-independent Course Data Bases:.....	29

Operation Examples	36
V. Implementation Problems	55
Learning the Target Machine and Languages.....	55
Checking the Graduation Requirements' Accuracy	56
Making Sure of the User Interface.....	57
Being Optimistic	58
Tracking Changes.....	59
Techniques Used.....	59
Documents Produced.....	60
VI. Future Use	64
Characteristics of the System.....	64
Suggestions for Future Improvements.....	65
Programming for Other Computer Systems.....	66
Appendices	
A. AUAS Proposal	
B. UM Catalog Requirements	

- C. Intermediate Transformed Requirements
- D. Final Transformed Requirements
- E. System Configuration
- F. Data Flow Diagram
- G. Data Dictionary
- H. Process Description
- I. File Description
- J. Structure Charts

List of Figures

1. AUAS Functions and Data Bases.....	9
2. AUAS Internal Program Structure.....	15
3. AUAS System	18
4. AUAS Life Cycle Chart	60

List of Tables

1. Transformation Comparison.....	25
2. ACCSD.DBF Course Data Base	32

I. Introduction

This chapter describes the University of Montana advising problems which students and advisers have faced for a long time and have not yet been able to solve satisfactorily.

Advising Problems

Currently, the University of Montana advising system is run by its departmental advisors. These advisors help students determine their graduation requirements so that they can graduate at the proper time.

When students present their transcripts and course request forms to the advisors, the advisors determine the requirements based on cross referencing information in transcripts and UM catalogs.

Advisors must be accurate when making decisions for the students. However, because the advisors are not trained especially to do advising, sometimes they make mistakes. The mistakes they make often are due to their unfamiliarity with the requirements.

Unfortunately, when the wrong information is given to the students, they are the ones who suffer. A student's graduation could be delayed or he could be required to take more courses than necessary.

Since the advisors' mistakes can not be completely avoided, there should be an easier way for the advisors to counsel the students so as to eliminate mistakes as far as possible. That is why an Automated Undergraduate Advising System (AUAS) was proposed - to decrease the advisors' mistakes, and to increase reliability of the advising system.

Proposed AUAS

The need for a reliable, easily maintainable computer advising system was proposed by Professor Randy Weirather from the Department of Communication Sciences and Disorders (CSD) to Professor Spencer Manlove from the Department of Computer Science (CS) in March, 1986.

After a feasibility study by the author, a formal proposal for the AUAS was made and approved by her committee members in April, 1986. AUAS is a prototype for use by the CSD Department. If possible, it will be adapted for other departments' use at a later time. In its present form, the system assists with the advising process for CSD students and their advisors. The original AUAS proposal is given in **Appendix A**.

System Goals

AUAS was designed to achieve the following goals:

- 1) To save the advisor's time.**

The system is designed to reduce the time an advisor spends in gathering graduation requirement data and calculating requirements. Therefore, by reading the requirement outputs made by the AUAS, an advisor can have more time to concentrate on his teaching and research duties.

- 2) To produce reliable requirement outputs whenever the correct information is provided.**

In other words, the AUAS is, like most software, a "garbage in, garbage out" system. The AUAS is programmed with built-in algorithms. It will work with correct data to produce reliable requirement outputs. Only programmers should be allowed to modify the algorithms. Thus, if data is given incorrectly by the user, the output will not make any sense. A meaningful output can be expected only if input data information is given correctly.

- 3) To be easy to operate, i.e. user friendly.**

With or without prior reading of the User's Manual, the AUAS's users should be able to use the system easily, depending only upon a personal computer background.

4) To be easy to maintain.

By reading the Technical Manual, future AUAS programmers should be able to maintain the system without changing the algorithms which have been built into it.

Graduation Requirements

There are seven important requirements in the catalogs (from 1982 until the current year) which are covered by the AUAS. The AUAS is built to handle the most difficult and tedious part of the advising process - cross referencing between the UM catalog and student data. By containing the most up-to-date requirement information and algorithms, the possibility of giving wrong information to the students should be reduced to a minimum.

II. System Overview

This chapter describes data bases, functions and program structure for AUAS.

AUAS Data Base Structure

There are three types of data bases used by AUAS programs, they are:

1) Student Data Bases

All student's data bases are uniquely identified by student's ID number, they include:

- A) The student's *Original Course Data Base* contains information on all courses which have been taken by the student. This data base serves as one of the major sources which helps AUAS to produce student's requirements; it will be modified by user as often as needed.

B) The student's *Selected Course Data Bases* are used by AUAS programs during processing of the student's advising results. These data bases will be displayed to the user only if some data has been selected by the system. In other words, a student or his advisor can control the contents of this data base; it is derived from the student's *Original Course Data Base* and AUAS programs' internal matching algorithms. Student's *Selected Course Data Bases* are uniquely identified by the student's ID number so that there will be no confusion between different student's *Selected Course Data Bases*. These include the following:

1) Selected Bachelor Degree Course Data Base stores a student's courses which have been taken based on the maximum credit applicable rules¹ for the bachelor degree.

2) Selected Minor Course Data Base stores a student's courses which have been taken from his minor department; this set will remain empty as long as the student has not declared a minor.

3) Selected Non-major Course Data Base stores a student's courses which have not been taken from his major department.

¹ The amount of credit which may be counted toward the minimum credit requirements for the bachelor degree is limited in certain areas; please refer to 1986 UM catalog, page 34 or Appendix B for the complete descriptions.

- C) The student's *Profile* data base contains each student's personal information: social security number, name, the quarter and year he entered UM, his major or minor and GPA (Grade Point Average), etc. This data base will also be modified by the advisor and the AUAS programs from time to time, when there is a need.
- D) The student's *Waived Course Data Base* contains the student's waived courses information, if applicable. Not every student has waived courses information, which indeed requires approval from the student's advisors. This data base will only be used as information for the advisor. The AUAS programs do not refer to this data base to produce the student's requirements because a waived course simply does not imply that the student has taken that course. Therefore, it should not be counted as one.

2) AUAS Data Base

AUAS data bases are only used by AUAS's programs; therefore, their contents will not be exposed to the user. They play very important roles in AUAS because they contain all its requirement data and requirement matching algorithms. For example, **ASRRB.DBF** is a 48 X 26 data base which stores all of the 48 catalog requirements; its data will be internally referred to by AUAS programs in producing the requirement results for students. More explanations are given in the following two chapters.

3) Course Data Bases

All Course data bases are, also, only used by AUAS's programs. They contain requirement course information which will be used to determine the fulfillment of student requirements. For example, **ACCSD4.DBF** is a course data base which stores all the required course information coming from the Department of Communication Sciences and Disorders core requirements in catalogs for all the years since 1982. This data base will then eventually be referred to by AUAS programs as a guide to tell whether or not a student has fulfilled the course requirement. More description will be given in Chapter IV, to show how the *Course Data Base* is used.

AUAS Functions

Following is an illustration of AUAS's functions:

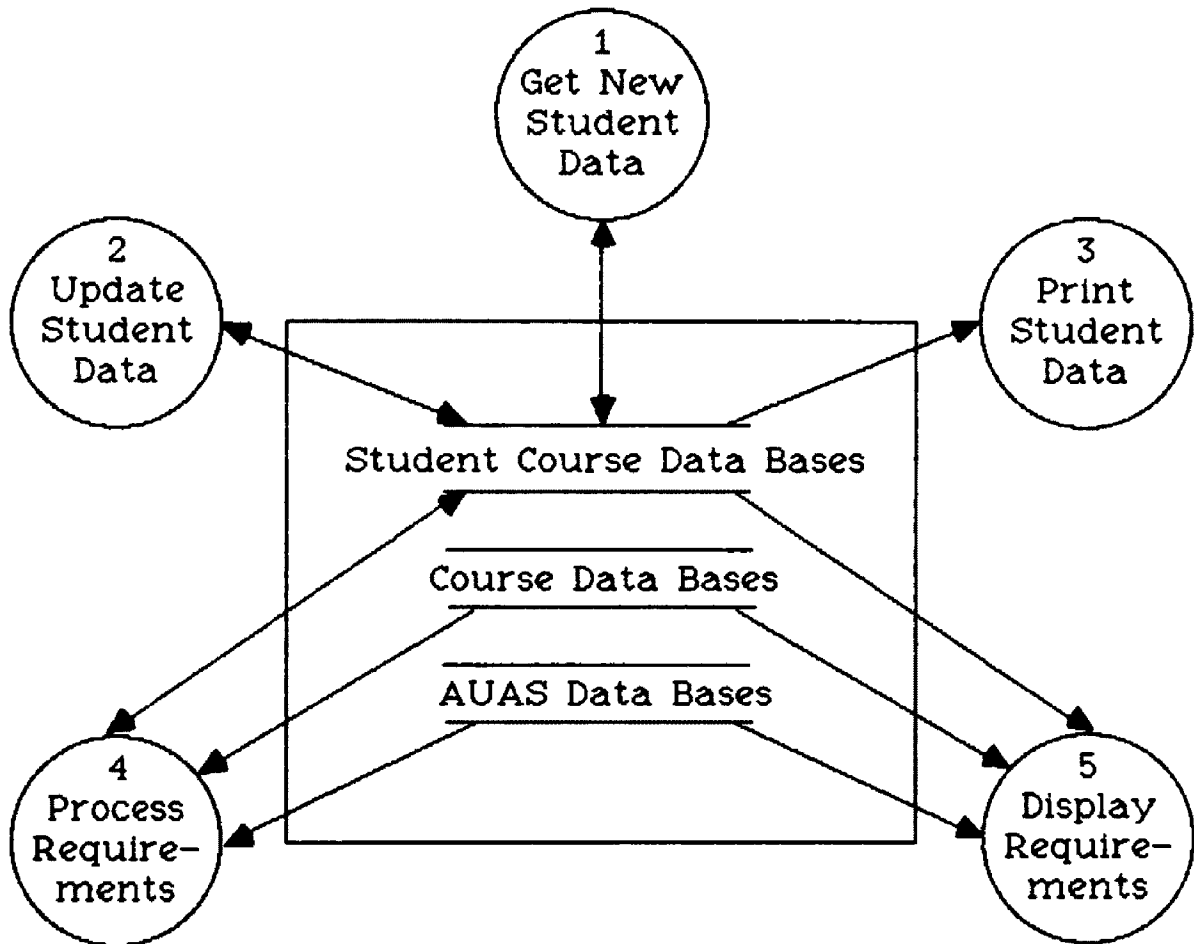


Figure 1. AUAS Functions and Data Bases

(Circles are "functions"; arrow lines represent data flows in and out of the functions; data bases are in the rectangular box.)

Each function has its own role in AUAS:

Function 1:

Obtains new student's ID

After a new student's ID is chosen by the user, all the data bases which are related to this student are prepared; then every other function's performance is based on this set of data.

The data bases which are used by this function are the student's, *1) Original Courses Data Base and 2) Profile Data Base*

Function 2:

Updates student data

This function allows users to update parts of a student's data bases, in particular the student's, *1) Original Course Data Base, 2) Waived Course Data Base, and 3) Profile Data Base.*

Function 3:

Prints student's data bases

This function allows user to print the following student data bases: *1) Original Course Data Base, 2) Waived Course Data Base and 3) Profile Data Base.*

Function 4:*Processes student's requirements:*

This function tells AUAS to produce requirement results based on the student's data bases which have been selected by *Function 1* or updated by *Function 2*. The process involves the following steps:

- a. Confirming that the student has taken or not taken the five exemption/tests which determine 5 of the possible 48 potential requirement results.
- b. Checking if function 4 has been run for this student before. If it has, then the system will re-use the previously *Selected Course Data Bases* based on the student's ID number to save process time. If it has not, the system will produce *Selected Course Data Bases* for the later use of AUAS programs.
- c. Prompting the advisor to decide if he does not want to run the rest of this function for complete new requirements to be produced. From this point, if the user has chosen to proceed, he should wait for the processing to be finished or he should return to choose other functions.
- d. Processing of each requirement based on previous information and all the AUAS data bases.

Function 5:

Displays student requirements

This function allows the user to view

- 1) the complete requirements on screen or printer.
- 2) some student *Selected Course Data Bases* which have been selected in function 4, if there are any.

AUAS Program Structure

The AUAS's capability will not be achieved without programs which have been built specially to fit the purpose of the system. Since AUAS is a highly user interactive menu driven system, it was designed according to its functions, which simply means that its programs are partitioned according to the above 5 functions.

In the following paragraphs we view the AUAS program structure from top to bottom based on control flows; they are *High-Level Control*, *Intermediate-Level Control*, and *Low-Level Execution* programs.

1. *High-Level Control* Programs

These programs are major controllers on the AUAS; they are simply menus themselves displayed in the screen. Each *High-Level control* program represents one of the 5 functions. Their control flows mainly correspond to the menu choices they present. Each *High-Level Control* program calls one to many *Intermediate-Level Control* programs according to the complexities of the function that the program performs.

2. *Intermediate-Level Control* Programs

These programs are called by *High-Level Control* programs; each *Intermediate-Level Control* program corresponds to one of *High-Level Control* program's menu choices. Each *Intermediate-Level Control* program calls many *Low-Level Execution* programs according to the complexities of the function that the program performs.

3. *Low-Level Execution* Programs

These programs are at the bottom line to serve their callers - *Intermediate-Level Control* Programs. They obtain data from AUAS's data bases, execute instructions and store the results in corresponding data bases.

The *Low-Level Execution* Programs are categorized into three different types with respect to the functions they perform; they are - *Front-End Matching* and *Output* processors. Whereas *Front-End* processors handle *Function 1, 2, 3* and part of *Function 4*, the *Matching* processor handles the remaining part of *Function 4*, and the *Output* Processor handles *Function 5*. These processors not only process the eligibility of students' requirement data contained in the AUAS's data bases² (including *student data bases, AUAS data bases, courses data bases*), but also produce the students' requirement outputs.

² The use of data bases and their relationships to programs will be explained in Chapter 5.

HLC = High-Level Control Program
 ILC = Intermediate-Level Control Program
 LLE = Low-Level Execution Program

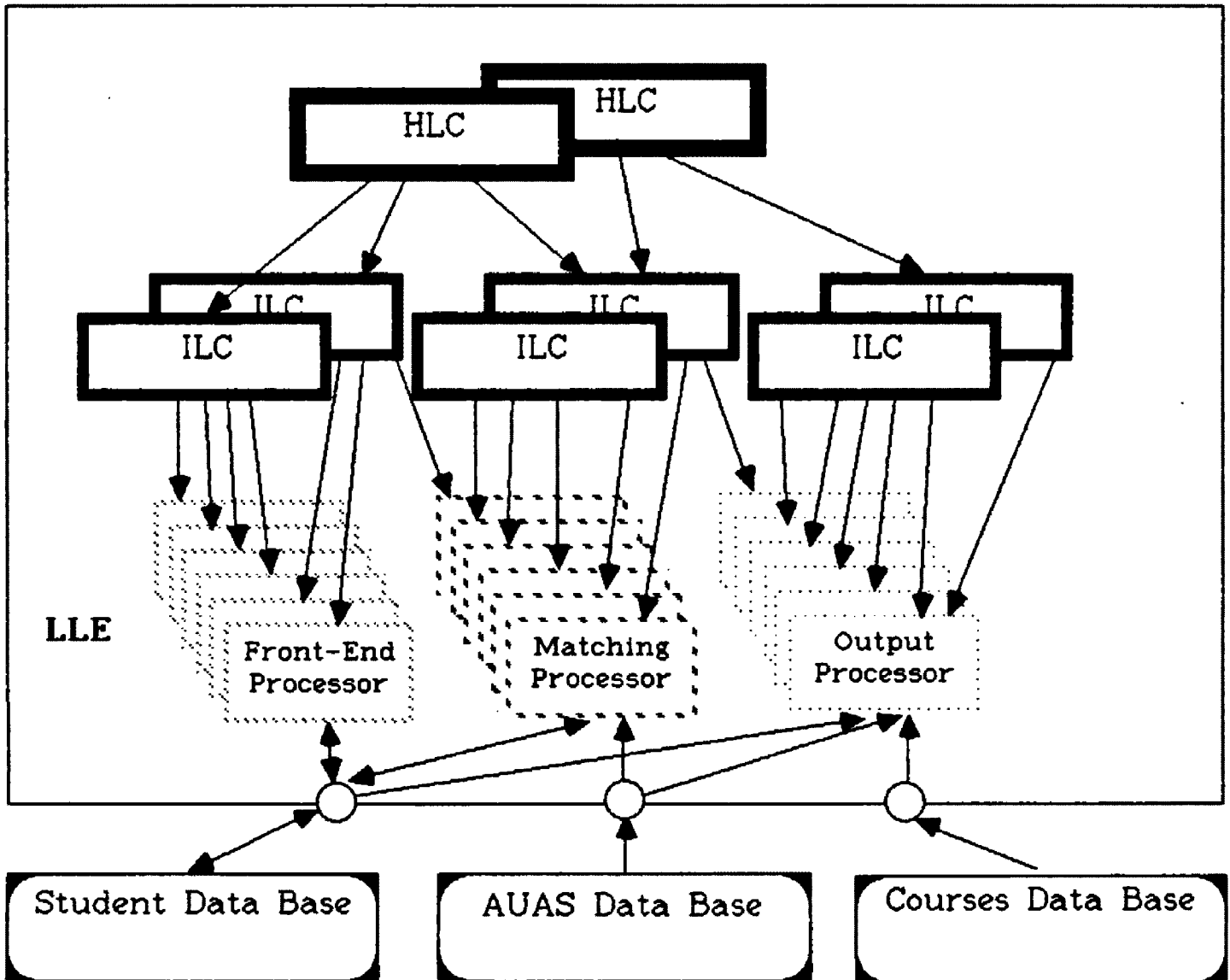


Figure 2 AUAS Internal Program Structure

(Arrow lines between HLC, ILC, and LLE programs are directions of control flows; arrow lines between LLE and Data Bases are directions of data flows. Circle nodes represent the access between programs and data bases. Among those three data bases, only the *Student data bases* can be modified by the programs - indicated by double arrow lines.)

Front-End Processor performs the following tasks:

1. Updating and validating the student's data bases.
2. Checking the student's *Profile* data which will be obtained by the *Matching Processor* later. For example, a student's *GER type* is a flag which indicates whether or not a student is allowed to choose *New or Old* General Education Requirements or both of them. A student's *GER type* is determined by the *Front-End Processor* based on the number of his transfer credits and the quarter and year he entered UM. His *GER type* will be used by the *Matching Processor* to select the applicable requirements for him.
3. Providing information to help an advisor understand the data he is getting.

Matching Processor performs the following tasks:

1. Obtaining data which has been pre-examined by the *Front-End Processor*.

2. Matching and cross referencing the data between *AUAS Data Bases* and *Student Data Bases* to produce requirements for the student.
3. Storing requirements in the student's output file which can be reached by the *Output Processor*.
4. Continuing to process requirements for the student's file until all the requirements have been examined.

Output Processor performs the following tasks:

1. Displaying a student's requirements from all the corresponding output files on either screen or printer, depending on the user's choice.
2. Displaying the information which will be based by AUAS to produce a student's requirements. For example, the advisor can read on the screen a brief description which determine a student's *GER type*

Among these three processors, the *Matching Processor* is the most important because it produces student's requirements. The following illustration summarizes AUAS's structure.

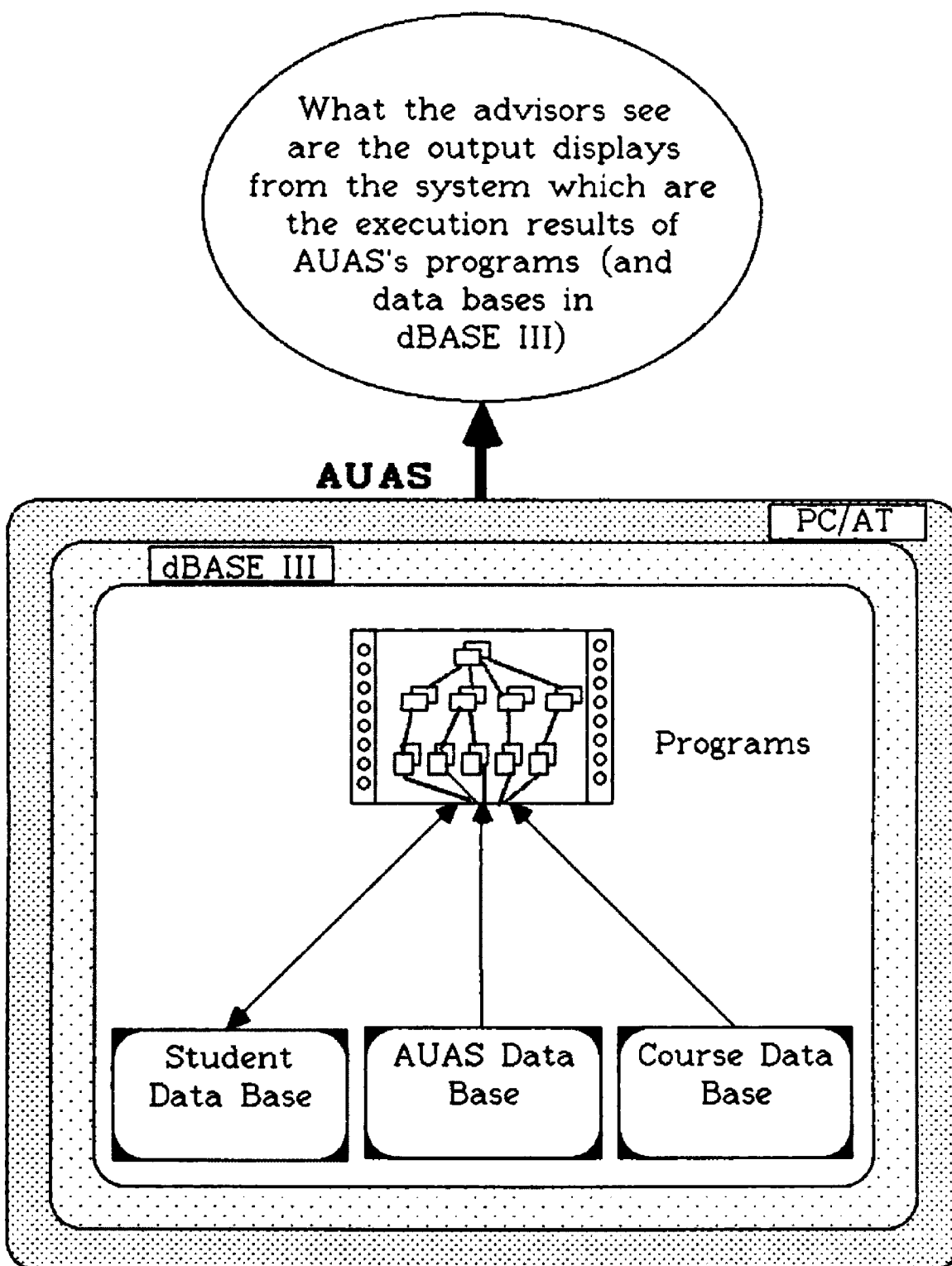


Figure 3 AUAS System

III. Implementation Method

This chapter describes the implementation method developed for solving problems of updating the AUAS requirements.

The Problem to be Solved

The biggest problem that the author faced was to transform the UM catalog requirements into AUAS data bases and programs. These programs, also called interpreters, interpret data obtained from the data bases, produce requirement results, and display the results. In order that no error should result from the transformation, every stage of the transformation was developed carefully to avoid its containing any misleading information. These stages¹ are described in the following sections.

The Transformation Operation

To transform the UM catalog requirements into useful AUAS data bases and programs successfully, a complete study² of them was made. **Appendix B** is included to show the original UM catalog requirements information³. The study result is shown in **Appendix C**.

¹ The results of the states are displayed in the Appendices B, C, and D.

² This study also included the CSD Departmental Requirements.

³ The requirements information was originally copied from the UM catalogs, from year 1982 until 1986.

In **Appendix C**, all the catalog requirements⁴ were re-written in a structured English which eliminated confusing and redundant phrases. What was left were well-defined, simplified requirements.

Based on the result given in **Appendix C**, a series of further transformations were conducted to form the AUAS programs and data bases. The transformations were:

Step 1:

Identify *key words* in each requirement so that every key word only has one specific meaning. Key words are defined as "important" words in a requirement, so important that the requirement will not make sense if its key words are left out.

Example - before step 1:

(the following requirement was copied from the 1986 Catalog, page 33)

Candidates for the Bachelor of Arts degree in the College of Arts and Sciences must complete a minimum of 93 credits in that College...

Example - after step 1:

(underlined words are key words)

⁴ The requirements processed by the AUAS are - Bachelor Degree, UM Writing Examination, UM Minor, Omnibus, Major-CSD Department, Old General Education, and New General Education requirements.

Candidates for the Bachelor of Arts degree in the College of Arts and Sciences must complete a minimum of 93 credits in that College...

Step 2:

After all the key words have been identified in Step 1, they are classified according to the function they perform in defining the requirement. These functions are given names; the relationship between a function's name and key words is given as follows:

1. Every function has one name and may have many different values. For instance, the function **TITLE** can have the value "BAR", Bachelor of Arts Degree Requirement, or "UMR", University Minor Requirement.
2. A function's values will be key words or abbreviated key words.

Example - same data as step 1

(**bold faced** words are function-names; the underlined word which appears in the same line with a function-name is the function's value)

A. Bachelor of Arts degree - function: **TITLE**;
this key word signifies a requirement's title

B. College of Arts and Sciences - function: **COURSE FILE TO BE MATCHED;**

this key word specifies the name of a course file that the student's courses taken may be checked against

C. 93 - function: **Required Number;**

this key word tells the advisor that 93 is the number of credits demanded by a requirement

D. credits - function: **Sum Type;**

this key word tells the advisor to accumulate the *credit* total of the courses which have been selected in **B.** and to display the result to show the fulfillment of this requirement by the student

After all function-names have been recognized in the requirements, we see that the same function-name may be referred to many times in different requirements. For example, these function-names - **Requirement Number**, **Sum Type** and **Course File to be Matched** are all recognized in the following three requirements:

Example - three of the CSD Departmental Requirements, 1986 Catalog, page 86.

(**bold faced** number - **Required Number** function
underlined word - **Sum Type** function
Italic word - **Course File to be Matched** function)

All students majoring in communication sciences and disorders are required to complete...:

Requirement 1:

A total of **31** credits in *Communication Sciences and Disorders..*

Requirement 2:

A total of **18** credits in *Psychology..*

Requirement 3:

A total of **5** credits in *Child Development..*

Step 3:

Create a two dimensional table in which to store the entire list of requirements. The number of rows is the total number of requirements which have been discovered (48 of them in this case). The number of columns is the number of different function-names recognized in Step 2 (26 of them have been found); every column is named by a unique function-name.

Step 4:

Fill in the table with the requirements from **Appendix C**. Process each requirement at one time by filling in the columns in the same row with the applicable function-names' values (key words or abbreviated key words). Ignore the columns which do not apply to that requirement.

The result is given in **Appendix D** (which is the ASRRB.DBF file). The differences between **Appendix B** and **D** are displayed in Table 1:

compared data	Appendix B (catalog information)	Appendix D (transformed information)
Matching Algorithms	implicitly embedded in the text	defined by the column entries, which are to be interpreted by the programs
Requirement Data	implicitly embedded in the text	stored in the columns, ready to be referred by the AUAS programs
Overall Usage	not much in the text, and that little is difficult to understand	enough included to be Data useful for the purpose of updating the AUAS programs and data bases

Table 1 Transformation Comparison

(Each column in **Appendix D** is actually an operation which operates on the data it contains. Therefore, a requirement in a row is defined by a group of column operations in that row.)

Transformed Requirement Data Base Structure

The ASRRB.DBF data base file is a 48 X 26 table. Notations defined as below are used throughout this chapter.

1. **Requirement_m** (m-1 to 48) represents:

The **m**th requirement (i.e. the **m**th row) appearing in the data base.

2. **Slot_{m,n}** (m-1 to 48, n-1 to 26) represents:

The data contained in a slot whose location is in the **m**th row and **n**th column of the data base.

As a matter of fact, most of the requirements in the data base require only a few of the 26 columns for their definition. The **Requirement_m** is defined by a group of meaningful and non-blank **Slot_{m,n}** (n- 1..26).

The reason for organizing the table in this manner is that a requirement can be easily updated with the changing catalog requirements.

For example, refer to one of the CSD Departmental Requirements examples in step 2 (other examples can be located in Appendix D from rows 11 to 16):

A total of <u>5 credits</u> in <i>Child Development</i> ..
--

The above requirement occupies three slots in the 12th row of the ASRRB.DBF- **Slot_{12.3}**, **Slot_{12.7}** and **Slot_{12.15}**. Later, the registrar may decide that this requirement should be changed to:

A total of **12 non-extension** credits in *Child Development*..

Notice that the number **12**, and the word **non-extension** are the only changes. In this case, the value of **Slot12.3** has been changed from **5** to **12**, and a new slot **Slot12.5** has been used to accommodate the newly added key word - **non-extension**.

Note that requirements may be updated by changing the contents of data base slots. However, if the 26 columns' functions⁵ do not fit the needs of the future, then the ASRRB.DBF's structure must be changed in order to meet the new requirements.

⁵ For a complete explanation of every column's function, please refer to the technical manual under the ASRRB.DBF section.

IV. Examples

This chapter provides additional examples of the operations related to AUAS data bases.

Understanding more about AUAS data bases

In addition to **ASRRB.DBF** (*one of AUAS Data Bases*), which has been mentioned in the last chapter, the *Student Course Data Base* and the *Course Data Base* (see Chapter II) were also built for the designated purposes.

A few examples will be given to explain the operations related with **ASRRB.DBF** and other data bases. Before we proceed, the following facts, on which the examples are based, should be understood.

1. A catalog year (also referred to as an academic year) starts in the **Autumn** quarter of that year, continuing to next year's **Winter**, **Spring** and **Summer** quarters. Therefore, 1985's catalog requirements will be in effect during Autumn 1985, Winter 1986, Spring 1986 and Summer 1986 quarters.
2. Any student who chooses to graduate under a specific catalog should fulfill all the requirements which are listed in that catalog.
3. The courses required may be changed in catalogs from year to year. This should not be of any concern to the students, however, because no matter how many changes have been made to the catalogs, the student is only responsible for fulfilling the requirements which are stated in his own chosen catalog.
4. In other words, if a student has taken courses which are not required by his chosen catalog, then those courses will not be counted towards his requirements' fulfillment. This may sound obvious, but students and advisors do have a tendency to forget this fact and actually count the non-eligible courses as part of the fulfillment.

Time-independent *Course Data Bases:*

The following paragraphs explain how the *Course Data Bases* help **ASRRB.DBF** to solve the problem of ongoing changes made in requirements and why they constitute time independent data bases.

If we take a closer look at **ASRRB.DBF**, we will find no column built for specifying the catalog year. How does **ASRRB.DBF** know which year catalog requirements are to be used in order to produce the correct requirement outputs for the students?

The answer is:

ASRRB.DBF does *not* need to know what catalog a student uses; it proceeds to handle every one of the 48 requirements stored in **ASRRB.DBF** which are applicable to a particular student. The requirement outputs produced will show that some requirements are demanded only if a student has chosen to graduate under a specific catalog year (this kind of information is stored in *AP1*, *AP2* or *AP3* columns in **ASRRB.DBF** as needed). That is, when the requirement does not specify a catalog year, that requirement is in effect every student in a particular category.

How does **ASRRB.DBF** manage to keep track of all course requirements in catalogs which, in fact, have been changed from year to year?

The answer is:

ASRRB.DBF, again, does *not* need to know what courses have been changed in the requirements; these are kept track of in the *Course Data Base*. They contain all the courses' *department codes, numbers* and the *catalog years* when the course was *first* required and *last* required (if applicable).

For example:

ACCSD4.DBF (one of the *Course Data Bases*) stores the departmental 400 level course requirements for the Department of Communication Sciences and Disorders (CSD). There are 11 such courses; information about them is given in **Table 2**, under the headings: **CD**-course department code, **CN**-course number, **CYF**-catalog year first required, **CYL**-catalog year last required (note that 83 catalog year means the 83-84 academic year):

record #	CD	CN	CYF	CYL
1	180	423		
2	180	431		
3	180	434		
4	180	435		
5	180	436		
6	180	442		
7	180	460		
8	180	477	84	
9	180	481	82	83
10	180	490		
11	180	495		

Table 2 ACCSD4.DBF Course Data Base

(180 is the department code for the Department of
Communication Sciences and Disorders.)

When AUAS programs see the above table, they interpret each of the
11 records in the following steps:

Interpretation Steps:

1. Check if there is any student's course whose department code and number are the same as **CD** and **CN**, if found, continue to step 2, else quit.

2. Check if **CYF** and **CYL** slots are both empty.

This means that there is no time limit set up for this course to be considered eligible; i.e., as long as this course is taken by the student, it will be counted towards this requirement. If so, select this course and go no further; else continue to step 3.

3. Check if **CYF** is not empty and **CYL** is empty.

This means that a course must be taken only *during or after* the catalog year indicated in order to be considered eligible since there is no data (**CYL**-empty) to check for a termination year. If true, select this course and go no further; else continue step 4.

For example, if **CYL-84**, then only if the student's course was taken after **Autumn 1984** (*not before* Autumn 1984) could it be counted towards this requirement.

The following table shows the possible quarters and years for a course to be counted towards this requirement (if its course department code and number have already been matched),
A-Autumn, W-Winter, S-Spring, Su-Summer quarters.

84 catalog year				85	86	...>>>>
84	85	85	85			as late
A	W	S	Su			as possible

4. Check if **CYF** is empty and **CYL** is not empty.

This means that a course must be taken *during or before* (i.e. no *later*) than the catalog year indicated in **CYL** in order to be eligible. If true, select this course and go no further; else continue to step 5.

For example, if **CYF**-84, then only if the student's course was taken during the 1984 catalog year, or before, can it be counted towards this requirement (*not after* Summer 1985 because the 1984 catalog year only involves **Autumn** 1984, **Winter** 1985, **Spring** 1985 and **Summer** 1985; therefore, **Summer** 1985 was the last quarter in catalog year 1984).

The following table shows the possible quarters and years for a course to be counted towards this requirement (if its course department code and number have already been matched),
A-Autumn, W-Winter, S-Spring, Su-Summer quarters.

...>>>>	82	83	84 catalog year			
as early as			84	85	85	85
possible			A	W	S	Su

5. The last possibility is that both **CYF** and **CYL** are not empty.

This means that a course must be taken

1) *during or after* the catalog year indicated in **CYF** *and*

2) *during or before* the catalog year indicated in **CYL**,

in order to be considered eligible. If true, select this course and go no further.

For example, if **CYF-82** and **CYF-83**, then only if the student's course was taken during the 1982 catalog year or after, **and** during the 1983 catalog year or before could it be counted towards this requirement.

The following table shows the possible quarters and years for a course to be counted towards this requirement (if its course department code and number have already been found matched), A-Autumn, W-Winter, S-Spring, Su-Summer quarters, bold faced-1982 catalog year, regular faced-1983 catalog year.

82	83	83	83	83	84	84	84
A	W	S	Su	A	W	S	Su

Notice that if course **CSD 481** (#9 in Table 2) was taken during the above time then it can be counted towards this requirement; but if it was taken in Autumn 1986, it will not be eligible because it was taken too late.

After all 11 records in **ACCSD4.DBF** have been processed by the above 5 matching mechanisms, a group of courses would be produced and they are eligible to be counted towards the CSD 400 level requirements. That is how **ASRRB.DBF** indirectly knows what courses have been required by catalogs throughout the years and how many of the student's courses can be counted towards the requirements.

Operation Examples

More examples are given here for describing the **ASRRB.DBF**'s column functions and the operations as they relate to the AUAS programs.

Example 1 - row #14 in **ASRRB.DBF** (Appendix D) which corresponds to the 1987 catalog, page 66:

Brief Introduction:

In the 1986 catalog, page 66, one of CSD's Major Credit requirements states that a CSD major student should take 5 credits in Physiology/Neurosciences.

(note that this requirement was only created after the 1986 catalog year)

Output Display:

The output display for this requirement is based on user's request (**X** will be the student's actual Physiology/Neurosciences credits total as calculated by AUAS programs, "5." means that this requirement is the fifth of 6 *CSD Major Credit Requirements*)

5. **X** of 5 required credits in Physiology/Neurosciences (this item is required beginning with the 1986 catalog).

Columns interpreted by AUAS:

The above requirement output results from the data in row #14.

The columns and their data are given by following:

column #	Column	Column's data
<i>2</i>	<i>Item</i>	<i>5</i>
<i>3</i>	<i>ReqNum</i>	<i>5</i>
<i>7</i>	<i>SmType</i>	<i>1</i>
<i>8</i>	<i>Oin</i>	<i>T</i>
<i>14</i>	<i>OP</i>	no value, empty string
<i>15</i>	<i>CF</i>	<i>ACPN</i>
<i>16</i>	<i>CS</i>	no value, empty string
<i>18</i>	<i>OSTP</i>	<i>0</i>
<i>24</i>	<i>API</i>	<i>(this item is required beginning with the 1986 cata</i>

25	AP2	log)
----	-----	------

The following are sequences of tasks performed by AUAS when it interprets the above information in **ASRRB.DBF**:

Task 1:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
15	CF	ACPN	<i>A Course Data Base file whose name is ACPN will be referred to in Task 2's operation</i>
16	CS	no value	<i>the Student's Original Course Data Base will be referred to in Task 2's operation</i>

Task 2:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
14	Op	no value	<i>selects courses from CS whose course department code and number can be found in CF, and saves them in a Temporary File</i>

Task 3:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
7	<i>SmType</i>	1	<i>Accumulates the credit total in the Temporary File, and saves this number in the student's Output File</i>

The credit total in the *Temporary File* will be the student's fulfillment number for this requirement. It will not matter if the student is not graduating under the 1986 catalog if this total is calculated anyway (remember that all output in the AUAS requirement results is used as reference. Therefore, by trying different possible catalog requirements, the output can help a student to realize how far he is from graduating.

Every student has his own *Outputfile* (identified by ID number) which stores the calculated *result* (if any result is calculated), *examination/test* result (if any is required by the requirement), *error* (indicator that this requirement is not applicable to the student). These are listed for each requirement processed in **ASRRB.DBF**. The result for *examination/test* and *error* will stay false until *EXAM*, *Stype* or *GER* columns have data in them; in this example there are no such results.

Task 4:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
<i>2</i>	<i>Item</i>	<i>5</i>	<i>Appends "5." to string S¹</i> <i>Appends credit total from student's</i> <i>Output file to S</i>
<i>3</i>	<i>ReqNum</i>	<i>5</i>	<i>Appends "of 5 required" to S</i>
<i>7</i>	<i>SmType</i>	<i>1</i>	<i>Appends "credits" to S</i>
<i>8</i>	<i>Oin</i>	<i>T</i>	<i>Appends "in" to S</i>
<i>15</i>	<i>CF</i>	<i>ACPN</i>	<i>Appends "Physiology/Neurosciences"</i> <i>S</i>
<i>24</i>	<i>AP1</i>	<i>(th...</i>	<i>Appends AP1's data to S</i>
<i>25</i>	<i>AP2</i>	<i>..)</i>	<i>Appends AP2's data to S</i>

Task 5:

Finally, adds *S* to student's Result File

¹ *S* is the string that will contain the output text.

Again, every student has his own *Result File* (identified by student's ID) which stores each requirement's output processed in **ASRRB.DBF** in the form of strings (string **S** from the above example). In order to save execution time, a student's *Result File* will be re-used many times (as long as the student does not add to his courses taken.)

A complete result string might be:

*5. 0 of 5 required credits in Physiology/Neurosciences
(this item is required beginning with the 1986
catalog).*

So far, a typical example has shown the sequence of interpretations accomplished in **ASRRB.DBF**. Some of **ASRRB.DBF**'s columns are interpreted before the others; for example, matching files (*OP*, *CF* and *CS*) comes before processing results (*OIN*, *ITEM* and *ReqNum*). Some columns are interpreted together (*OP*, *CF*, and *CS*) to make a course selection; others are simply output indicators (*OIN* indicates that the string "in" is to be included in the output.)

Example 2 - row #17 in ASRRB.DBF (Appendix D) which corresponds to the 1987 catalog, page 66.

Brief Introduction:

In 1986 catalog, page 66, one of CSD's Core Course requirements states that a CSD major student should take 14 credits in CSD core courses at the 400 level or above.

(note that this requirement has been in force since the 1982 catalog.)

Output Display:

The output display for this requirement is based on a user's request (**X** will be the student's **deficient credits total** as calculated by AUAS programs, "2." means that this requirement is the second of 3 CSD Core Course requirements.)

2. **X** of 14 required credits in CSD, 400 level or above, have yet to be completed. This deficiency may be filled by the following courses:

<u>Dept</u>	<u>Number</u>	<u>Title</u>	<u>Which is required by catalogs</u>
180	431	...	on going
180	442	...	on going

....

(depends on the deficient course contents)

Special user request for the above requirement:

Notice that the catalog requirement does not say how to present the student's fulfillment or deficiency of this requirement; therefore, the advisor may decide that he wants to see the "deficient" credit total and the potential "deficient" courses which would help him and the student to survey what the course options are to fulfill the requirement.

Columns interpreted by AUAS:

The above requirement output results from the data in row #17.

The columns and their data are given in following table:

column #	Column	Column's data
<i>2</i>	<i>Item</i>	<i>2</i>
<i>3</i>	<i>ReqNum</i>	<i>14</i>
<i>8</i>	<i>Oin</i>	<i>T</i>
<i>9</i>	<i>CD</i>	<i>180</i>
<i>10</i>	<i>OD</i>	<i>F</i>
<i>11</i>	<i>GL</i>	<i>>=</i>
<i>12</i>	<i>CN</i>	<i>400</i>
<i>14</i>	<i>OP</i>	<i>2</i>
<i>15</i>	<i>CF</i>	<i>ACCSD4</i>
<i>16</i>	<i>CS</i>	no value, empty string

<i>17</i>	<i>OSS</i>	<i>A1</i>
<i>18</i>	<i>OSTP</i>	<i>1</i>
<i>24</i>	<i>AP1</i>	<i>, have yet to be completed. This deficiency may be</i>
<i>25</i>	<i>AP2</i>	<i>filled by the following courses:</i>

The following is the sequence of tasks performed by AUAS when it interprets the above information from **ASRRB.DBF**:

Task 1:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
<i>15</i>	<i>CF</i>	<i>ACCSD4</i>	<i>A Course Data Base file whose name is ACCSD4 will be referred to in Task 2's operation</i>
<i>16</i>	<i>CS</i>	<i>no value</i>	<i>the Student's Original Course Data Base will be referred to in Task 2's operation</i>
<i>17</i>	<i>OSS</i>	<i>A1</i>	<i>A file whose name is A1 will be referred to in Task 2's operation</i>

Task 2:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
<i>14</i>	<i>Op</i>	<i>2</i>	<p><i>1. selects the courses from CF whose course department code and number can not be found in CS, and saves them in OSS (A1 File)</i></p> <p><i>2. selects the courses from CS whose course department code and number can be found in CF, and saves them in a second Temporary File</i></p> <p><i>3. calculates the credit total in the second Temporary File and saves this number in student's Output File</i></p>

Task 3:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
<i>3</i>	<i>ReqNum</i>	<i>14</i>	<p><i>Subtracts the credit total in the OSS (A1 file) from 14, and saves this result in student's Output file</i></p>

Task 4:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
<i>2</i>	<i>Item</i>	<i>2</i>	<i>Appends "2." to S</i> <i>Appends credit total obtained from student's output file to S</i>
<i>3</i>	<i>ReqNum</i>	<i>14</i>	<i>Appends "of 14 required" to S</i>
<i>8</i>	<i>Oin</i>	<i>T</i>	<i>Appends "in" to S</i>
<i>10</i>	<i>OD</i>	<i>F</i>	<i>flag indicates to append CD's abbreviated department name</i>
<i>9</i>	<i>CD</i>	<i>180</i>	<i>Appends "CSD" to S</i>
<i>11</i>	<i>GL</i>	<i>>=</i>	<i>flag indicates that "level or above," should be appended before CN</i>
<i>12</i>	<i>CN</i>	<i>400</i>	<i>Appends ", 400 level or above," to S</i>
<i>24</i>	<i>AP1</i>	<i>(th...</i>	<i>Appends AP1's data to S</i>
<i>25</i>	<i>AP2</i>	<i>..)</i>	<i>Appendix AP2's data to S</i>

Task 5:

Adds **S** to Student's Result File

Task 6:

<i>Col #</i>	<i>Column</i>	<i>Data</i>	<i>Action</i>
<i>17</i>	<i>OSS</i>	<i>A1</i>	<i>displays A1 in the style indicated in OSTP</i>
<i>18</i>	<i>OSTP</i>	<i>1</i>	<i>displays requirement course's department code, name, number and the time when that course is required by catalog</i>

If the student has completed 2 out of 14 required credits from *ACCSD4*, then **2** will be the credit total, and the complete result string will be:

2. 2 of 14 required credits in CSD, 400 level or above, have yet to be completed. This deficiency may be filled by the following courses:

(followed are the contents of un-matched ACCSD4.DBF courses which were saved in A1; therefore, A1 is to be displayed here)

<u>Dent</u>	<u>Numb</u>	<u>Title</u>	<u>Which is required by catalogs</u>
180	431	Stuttering	on going
180	436	Vocal Behav	on going
180	442	Diag Appr	on going
180	477	Hab & Reba	after '84
180	481	Clinc Aud	between '82-'83

From the above examples, it should be clear that the more "user-flavored" the requirement output style is, the more sophisticated the interpretation processes becomes. Fortunately, there are not too many such cases. If special cases like the above are to be added later to AUAS, then there will be a potential problem; too many special cases will not only slow down execution time but they will also increase the burden of updating the **ASRRB.DBF** data base and the AUAS programs.

Example 3 - row #32 in **ASRRB.DBF** (Appendix D) which corresponds to the 1987 catalog, page 31:

Brief Introduction:

In the 1986 catalog, page 31, one of the 6 perspectives in General Education Distribution Requirements (GER) requires that a student should take 6 credits in Expressive Arts if his student type is 1 or 2.

Note that a student's type is determined by the number of transfer credits he carried when he entered UM, i.e.,

with transfer credits 0 to 45, he belongs to type 1

with transfer credits 46 to 90, he belongs to type 2

with transfer credits over 90, he belongs to type 3

(Notice also that this requirement was only imposed after Autumn 1984; it is one of the "New" General Education Requirements. Therefore this requirement must be fulfilled by students who choose to graduate after Autumn 1986.)

Output Display:

The output display for a requirement, if the student is eligible to apply, is also based on the user's request (**X** will be the student's actual Expressive Arts credit total calculated by AUAS programs, "1." means that this requirement is the first of 6 New GER Distribution requirements.)

New GER Distribution Requirements
--

1. X of 6 required credits in Expressive Arts.

Special user request:

Notice that there are two sets of demands in the New General Education Distribution Requirements (one set is only applicable for students of type 1 or 2, the other is only applicable to students of type 3.) Since only one of the two requirements will make sense if the student's type satisfies the condition, he may not want to see any non-applicable requirements. The AUAS has to make sure in this case not to display any non-applicable requirements.

Columns interpreted by AUAS:

The above requirement output is obtained from the data in row #32. The columns and their data are given in following table:

column #	Column	Column's data
<i>1</i>	<i>Title</i>	<i>GD</i>
<i>2</i>	<i>Item</i>	<i>1</i>
<i>3</i>	<i>ReqNum</i>	<i>6</i>
<i>7</i>	<i>SmType</i>	<i>1</i>
<i>8</i>	<i>Oin</i>	<i>T</i>
<i>13</i>	<i>CL</i>	<i>A</i>
<i>19</i>	<i>GER</i>	<i>N</i>
<i>22</i>	<i>Stype</i>	<i>12</i>

The following is the sequence of tasks performed by AUAS when it receives the above information from **ASRRB.DBF**:

Task 1:

Col #	Column	Data	Action
<i>19</i>	<i>GER</i>	<i>N</i>	<i>Checks the student's profile to see if he has chosen to use New or OLD GER.</i>
<i>22</i>	<i>Stype</i>	<i>12</i>	<i>Checks the student's profile to see if his student type is "1" or "2"</i>

If either of the above is false, then the student will not be eligible to apply this requirement, so continue to task 2.

Task 2:

<i>13</i>	<i>CL²</i>	<i>A</i>	<i>Searches for all the student's courses whose Course-Letter (CL) value is equal to "A", saves the matched student courses in a Temporary file</i>
-----------	-----------------------	----------	---

There are 9 types of *Course-Letters* which reflect the different regions of General Education Requirements. They are:

- h:** Humanities and the Arts
- b:** Behavioral and Social Sciences
- s:** Sciences and Mathematics
- A:** Expressive Arts
- E:** Ethical and Human Values
- H:** Historical and Cultural Studies
- L:** Literary and Artistic Studies
- N:** Natural Sciences
- S:** Social Sciences

² See next page for definition of *Course-Letter*.

Every course will be assigned a (prefix or suffix) *Course-Letter* if that course was approved by the registrar's office as a General Education Requirement course at the time when a student took it. Therefore, collecting all the student's courses lettered "A" will give the eligible courses towards this requirement.

Task 3:

<i>7</i>	<i>SmType</i>	<i>1</i>	<i>Calculates the course credit total in the temporary file, saves the result in the student's Output file.</i>
----------	---------------	----------	---

Task 4:

<i>1</i>	<i>Title</i>	<i>GD</i>	<i>Appends "New GER Requirements" to S</i>
<i>2</i>	<i>Item</i>	<i>1</i>	<i>Appends "1." to S</i>
<i>7</i>	<i>SmType</i>	<i>1</i>	<i>Appends the result from output file (from Task 3)</i>
<i>3</i>	<i>ReqNum</i>	<i>6</i>	<i>Appends "of 6 required" to S</i>
<i>8</i>	<i>Oin</i>	<i>T</i>	<i>Appends "in" to S</i>
<i>13</i>	<i>CL</i>	<i>A</i>	<i>Appends "Expressive Arts" to S</i>

If the student has not completed the required credits from *Expressive Arts* then **0** will be the credit total, and a complete result string will be:

New GER Distribution Requirements

1. 0 of 6 required credits in Expressive Arts.

As a result of the above three examples, we should realize that every column in **ASRRB.DBF** has specific functions associated with the data it carries. A column is considered to be highly valuable if its function is multi-purpose; (e.g., *CF*, *CScan* can be used in matching and assembling output strings). In other words, a column does not have much usefulness if there is only one function in it (e.g., *Title* is only used for displaying a requirement's title name).

ASRRB.DBF is not fixed to be a 48 X 26 table; it can be either shrunk or expanded according to the potential needs of user and programmer. The interpretation for each row and column may not always be fixed as they are now; it will be the programmer's responsibility to enhance or simplify any column's functions.

It is suggested that anyone who is interested in following up AUAS should study the Technical Manual in order to understand the actual interpretation algorithms. Although studying someone else's codes or technical material may not be an easy job, it will pay off if the goal is to fully understand the functions of **ASRRB.DBF**.

V. Implementation Problems

This chapter systematically describes the AUAS development process, the problems encountered and their solutions.

Learning the Target Machine and Languages

The AUAS target machine was determined to be the IBM AT personal computer (or compatible machine), since the CSD department wishes to use its own PC system instead of the university main-frame computer.

The next step was to decide the programming language to be used; dBASE III data base language was chosen because of its unique ability to handle data bases.

Although the Golden Rule¹ does not favor being aware of the physical device (machine) and the programming language, it must, however, be modified to fit the special AUAS case. That is, the AUAS was developed by a "One-Man Team." Therefore, knowing more about the machine and language to be used actually helped the development process.

¹ **Machine and language considerations should be independent of the requirements and design.**

Checking the Graduation Requirements' Accuracy

Since UM catalogs are the only information source² on which to base AUAS requirements, the accuracy and meaning of the catalog requirements is therefore very important.

Every requirement rule in the catalog (among the seven major requirements) was carefully interpreted and defined by the author in cooperation with the authorities to make sure that the requirements were understood correctly. This helped to ensure AUAS reliability.

Consulting with the administration took time. Since the UM catalogs have been used for a long time, and no one seems to have questioned them before, it often took quite a while to come up with answers to questions about them.

Unfortunately, the process of solving every dispute was on-going throughout the AUAS life cycle. It did prolong AUAS's development time, since all the questions of meaning should have been taken care of at the beginning of the "Requirement Specification Phase," instead of in the later phases.

² All the AUAS graduation requirement data are obtained from the UM catalogs.

It was surprising to learn that even though the UM catalog requirements are stated as clearly as possible, they are still unclear to some advisors and students. This is despite the registrar's opinion³ that just carefully reading the requirements will guarantee full understanding of them.

Once the catalog graduation requirements were finally established, a user interface prototype was developed.

Making Sure of the User Interface

The user interface prototype was used to illustrate the AUAS at the end of the Requirement Specification phase. For the author, the prototype was used to get the input and output interface information for the AUAS. For the user, it served as a channel which allowed the user and the author to communicate. The AUAS prototype merely served as an input-output scratch paper. This scratch paper defined the input-output functions for both user and author.

When the AUAS prototype was made, it was not made to correspond with the programming language's capability. The finished system may not, or would not, be exactly the same as the prototype system, even though the functions of the prototype remain the same.

³ Based on conversation with a registrar.

However, the user did not understand that there were differences between the final and prototype AUAS system. That was why when the finished AUAS system was presented to the user, questions were asked by him such as, "I thought it was supposed to look like the paper system you showed me before"... "The format looks different now"... "Paragraph indentation, bold face headlines were all gone, etc...".

In any case, changes were made to satisfy the user's expectation of the system. Details like this should have been thought of and avoided when the requirement specifications were done.

Being Optimistic

It would have been impossible for the author to complete the AUAS without being an optimistic person. Numerous changes were made on a daily basis, and it seemed that no matter how many changes had already been made, there were still changes to be made later.

As the author recalls, there was nothing done incorrectly in the life cycle phases. Why then so many changes in the final state of the AUAS? Maybe "perfectionism" is the reason.

Tracking Changes

When developing such a large system for a personal computer, every change made in the system should be tracked. Several techniques were created to keep track of the updating process. For example, all the AUAS program names and their brief descriptions were recorded in a file according to their calling sequences. A programmer can update this file if there is any calling sequences changed among these programs. The same scheme can be also applied to AUAS data bases.

Techniques Used

Software Engineering methodologies were used throughout the AUAS life cycle. It took more than one year to finish the AUAS. What follows is a summary of the time spent in the AUAS life cycle phases.

March - April -----Feasibility Study, Proposal
(1 month)

April - July -----Specifications (3 months)

July - October -----Design (3 months)

October - February -----Coding (4 months)

February - July -----Testing (5 months)

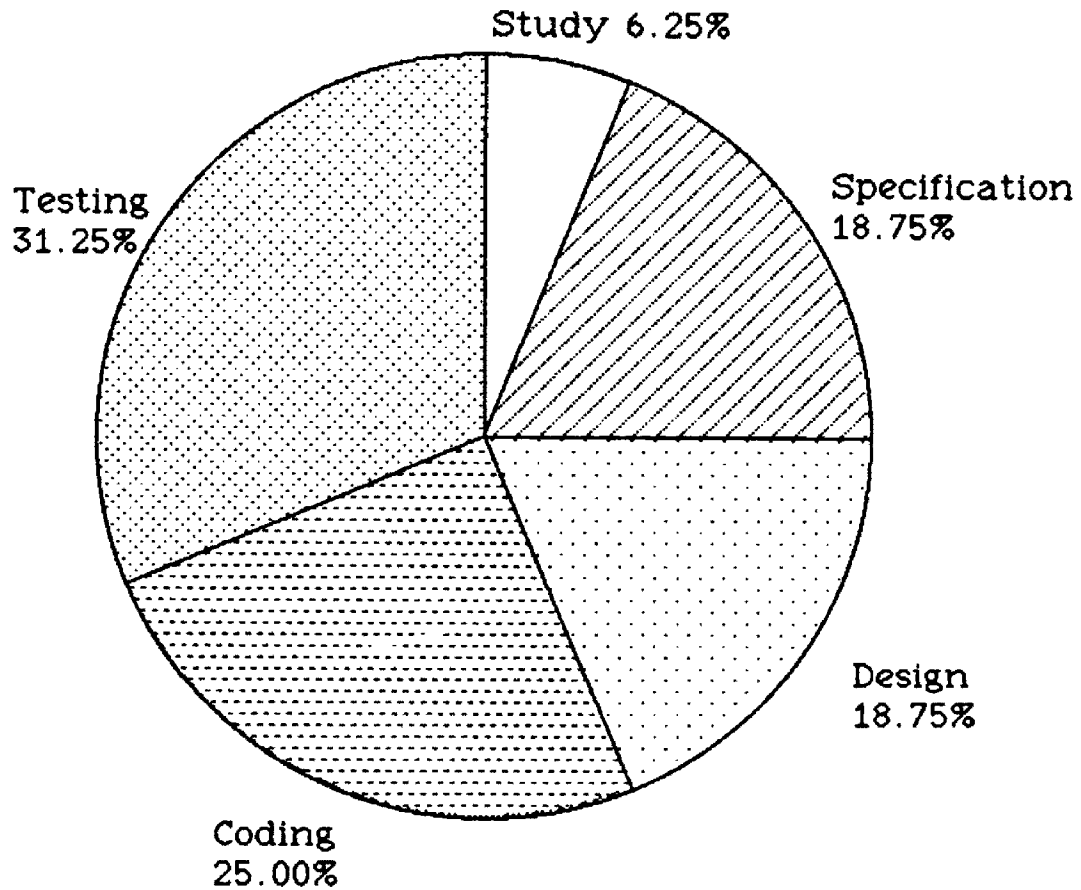


Figure 4 AUAS Life Cycle Chart

It is difficult to recall the exact dates when development of one phase allowed moving on to the next phase. Phase activities overlapped between neighboring phases. Corrections made in one phase might cause other phases to be updated. Phases were gradually finished when all the previous and current problems were solved.

Documents Produced

Documents were produced throughout the Life Cycle. They are:

Proposal (Appendix A)**Requirement Specification⁴**

Current System Study

Future System Study

Transformation Procedure

Data Flow Diagrams (Appendix F)

Data Dictionary (Appendix G)

Process Description (Appendix H)

File Description (Appendix I)

Prototype Menus for AUAS

Design⁵

Structured Charts (Appendix J)

Module Descriptions

First version of AUAS

⁴ This document specifies the structure analysis requirements for the AUAS. It is stored in the Department of Computer Science.

⁵ This document specifies the structure design for the AUAS. It is also stored in the Department of Computer Science.

Software Disk⁶**User Manual⁷**

Screen Menu Samples

Technical Manual⁸**Codes⁹**

AUAS Programs

AUAS Data Base Files

Utility Programs

Test Data

Output Samples (Screen Output and Printout Samples)

-
- 6** The entire AUAS is already installed in the CSD Department's PC hard disk. There are three floppy disks used for storing the AUAS programs and requirement data bases. These three disks are stored in the CS Department.
- 7** The user manual is made for the AUAS users. It contains the instructions for using the AUAS. This document is kept in the CSD Department.
- 8** A technical manual was made for the follow-up programmers who will maintain the AUAS. It contains complete introductions to the AUAS programs and data base structures. This document is also stored in the CS Department.
- 9** This document is included in the Technical Manual.

Due to the large volume of documents, they will not be attached to this professional paper (except for the **AUAS Proposal, Data Flow Diagrams, Data Dictionary, Process Description, File Description, and Structure Design**). The rest of the documents are stored either in the CS or CSD Departments as indicated in the footnotes. It is recommended that any interested person who wishes to study the AUAS should check with the above departments.

VI. Future Use

Hopefully, the AUAS can be maintained to meet requirement changes made in future catalogs. This chapter summarizes the AUAS characteristics and suggests possible future improvements and alternate ways to maintain the AUAS.

Characteristics of the System

Some of the characteristics of the AUAS are:

1. Accuracy:

This was the most important goal for the AUAS. Every piece of data stored in the AUAS data bases is examined to ensure reliability of the advising result.

2. Ease of updating:

The requirement data bases and programs can be easily updated through the help of the Technical Manual or the algorithms developed in the data structures.

3. User Friendliness:

It is a useable and user friendly system!

4. Use of color displays on screen:

While the user is waiting for the requirement results to be processed, the screen advises him with colored messages.

Throughout the entire operation of the system, the user will always find that specially designed color messages are displayed to help him to distinguish between the more important messages and to enter the right choices easily. Also, flashing colors are used to indicate the most important messages.

Suggestions for Future Improvements

There are several possibilities for future improvements of the system.

They are:

1. Expert System:

An Expert Advising System would not only perform the AUAS's current function but also help to make the adviser's decisions.

2. Faster System:

The current speed for AUAS to process the requirement results for a student who has taken 50 courses is about half an hour. This is because the dBASE III programming language must frequently recall information on disk. Alternatives may be sought by using different programming languages to implement AUAS.

Programming for Other Computer Systems

Apparently, the current computer system (IBM PC) will not have the computing and memory power to achieve future improvement goals. A main frame computer system which can handle more sophisticated situations should be considered.

The length of software's lifetime depends on its flexibility and durability. All software will eventually be phased out. By maintaining the AUAS to meet requirement changes, its lifetime can be prolonged.

The author believes that the AUAS is a good prototype which can be actually used as an example for a university-wide project¹.

¹ Commercially made software is currently being sought by the registrar's office to handle university-wide advising.

Appendix A

AUAS Proposal

Introduction:

The University of Montana maintains that academic advising is an important part of the educational process. Academic advising is required each quarter for all undergraduate students. Although faculty advisors are there to help, it is the responsibility of students to meet all graduation requirements. The UM catalog is the official source of information on these matters. The following paragraphs briefly describe the requirements and the advising process.

The Requirements**• The Degree Requirements**

A student may graduate by fulfilling university and departmental requirements in any University of Montana catalog under which she or he has been enrolled during the six years prior to graduation. The student may meet major and minor requirements under different catalogs than the catalog under which she or he is meeting university requirements.

• The Bachelor's Degree Requirements

A total of 195 credits is required for graduation with a bachelor degree. Of these 195, no more than 69 can be in the major field.

• **The Major Requirements**

Students must complete a major requirement in order to earn a bachelor's degree. The major department decides all catalog major requirements.

• **The Minor Requirements**

Students may elect to complete one or more minors in fields outside their majors.

• **The GER Requirements**

To receive a bachelor's degree all students must complete the General Education Requirement course requirements listed in the catalog.

The Advising Process

• **Registration:**

To finish registration, the student must get his proposed course schedule approved by his advisor. To help the advisor approve the schedule, the student must follow the following steps.

• Submit Student Information:

Generally, a student's information is kept by his department. This includes his academic record (transcripts, letters of recommendation, GRE test scores, etc.), and his admission status data (freshman, transfer, transfer credits, etc.). To start the advising process, the student may get his own information file from his major department and bring it to his advisor.

• Decision making:

The advisor will read the student information and try to understand the student's background. Besides this, the advisor also uses a checklist of GER and departmental requirements to track the student's progress and project the courses needed for upcoming quarters. Based on the above examination, the student's course schedule will be revised and signed by the advisor.

The Advising Problem:

We all know that it is difficult to understand the information concerning a student in a short time. Besides, there are too many things to be tracked and updated concerning a student's needs. Different students have different backgrounds (transfer credits, exceptions, etc.); therefore, different advising schemes are applied to them.

Quarter after quarter, the catalog requirements may be changed. Because different students need different types of advising, the advisor has to make his best effort to "match" the student's needs and the requirements' needs. We know that this is time-consuming, inefficient, and error-prone.

We can not prevent the catalog requirements from being changed, but we can prevent or reduce human errors with a computer's help. A computer system which helps the advising process is proposed in the following sections.

The Proposed System:

Currently the University of Montana Department of Communication Sciences and Disorders wants to automate its undergraduate student advising system. The AUAS (Automated Undergraduate Advising System) will help the advisors perform the following tasks:

1. Update each student's advising record which consists of:
 - Student information
 - Courses completed
2. Update a database of the catalog requirements which are:
 1. Bachelor Degree Requirements
 2. General Education Requirements
 3. Departmental Major Requirements
 4. Minor Requirements
 5. Other Requirements

3. Calculate the student's completed credits by matching his advising record with the requirements, which includes:
Adding cumulative credits obtained towards each individual requirement, and displaying the sum to the terminal screen/printer.

In order to be easy to use, the AUAS will be a menu-driven system. The users will be recognized by the AUAS password examination. The major functions of AUAS will be :

1. Query on-line data base of student's information. Besides, based on different query questions, all of the above information can be cross referenced to produce new answers.
2. Update student's information and requirements information.
3. Print student's advising result.

The Environment/Goals of the Proposed System:

The proposed AUAS is to be implemented in the IBM Personal Computer AT or compatible system using the Data Base Programming Language dBASE III. The AUAS should have as long term goal:

To serve as a possible model which the Administration could use to build a completely automated advising system for the whole university.

and the following quality characteristics:

1. User-friendly:

The man-machine interfaces should be user-friendly, so that the AUAS will be easy to use. The user manual will help users to use the AUAS without being exposed to the AUAS's internal functions.

2. Maintainable:

The AUAS should be easy to maintain. The technical manual will help the technical users (programmers) to maintain the performance of the AUAS.

Academic Benefits:

The following are the academic benefits from accomplishing the proposed system.

1. To learn software engineering methodologies more thoroughly by applying them.
2. To learn different ways of applying rule-based algorithms to data based inferences.
3. To learn to work with a real world user/user group. These communication skills are sometimes neglected by a project manager, which could lead to the failure of the project.
4. To learn to design a reliable system which acts like a fool proof black box for the users and is well documented internally for the programmers - the ideal goal for any software engineer.

General Benefits:

The following are the general benefits to users from implementing the proposed system.

1. To have a reliable software system to help the advising process for the CSD Department.
2. To save time on advising. Since the automated system searches, calculates and updates all of the student's information and requirements, the time saved for advising can be used for other "heuristic" advising.
3. To make less mistakes. The advisor and the student can concentrate more on the "heuristic" part of the advising - the abilities, the interests, the liabilities, so that mistakes in advising will be minimized.
4. To save effort! By spending less effort and less time with each student, the advisor can give more attention to other things. The computer can offer more powerful and efficient ways to access information, and the advisor can then afford a more graceful and delightful attitude toward student needs.

The Proposal's Goal:

The goal of this proposal is to get permission to proceed with the proposed system as a master thesis project. The extensive study already done of the proposed system has shown that it will not be trivial or simple. Over the long term, I would like to be a computer science graduate student who really accomplishes something for her campus by dealing with a real "need and benefit" situation.

The Proposed Development Schedule:

The following development schedule proposes deadlines for each phase of the software life cycle during the year of 1986.

1. Proposal, Software Plan, System Description By May 7
2. Requirement Analysis & Specification By May 25
3. Design By June 30
4. Coding By July 30
5. Testing, Testing Plan By August 5
6. Deliver By August 10

Appendix B

UM Catalog Requirements

refer to 1986-87 University of Montana Catalog

page 31 - 39 and 66 - 69

Appendix C

Intermediate Transformed Requirements

Notations used in the following paragraphs are:

1. Italic, bold faced lines indicate *Requirement Title*
2. Requirements are grouped together under the same Requirement Title.
3. Underlined blank words - ' _____ ' are to be filled by a number indicating which calculation result is to be used from the advising process.
4. Small boxed prints are explanations of the conditions applicable for the requirement they serve.

BACHELOR DEGREE REQUIREMENTS

The student has completed the following toward fulfillment of the Bachelor Degree Requirements:

1. _____ of 195 required credits.
2. _____ of 93 required credits in the College of Arts and Sciences.
3. _____ of 45 required UM(non-transfer) credits.
4. _____ of 30 required UM(non-transfer) credits among the last 45 credits earned.(this criterion applies to graduating seniors only; National Exchange Students may be exempted)
5. _____ of 45 required UM Non-Extension credits (for a second BA degree only).

A student must fulfill the above first three - item 1, 2 and 3 requirements to earn a second BA degree: item 4 and 5 only apply to different conditions - refer to 1986 Catalog, page 32.

UNIVERSITY WRITING EXAMINATION REQUIREMENTS

1. _____ credits must still be earned prior to eligibility for the University Writing Examination.

A student who entered UM after Autumn 1985 has to take at least 100 credits and pass one Writing (W) course before being eligible to take the Writing Examination.

A student entered UM in Autumn 1985 or after has to take at least 96 credits and pass one Writing (W) course before being eligible to take the Writing Examination - refer to 1986 Catalog, page 37.

UNIVERSITY MINOR REQUIREMENTS

1. _____ of a required 24 credits must have been earned from the minor department.
2. _____ of a required 12 UM credits must have been earned in the minor field.

A student can choose more than one minors. Each minor must be one of the approved University Departments - refer to 1986 Catalog, page 19-20, page 39.

OMNIBUS REQUIREMENTS

1. _____ of a maximum 40 credits for a B.A. must have been earned under the Omnibus option for independent work (Note: A maximum of 15 per topic is allowed).

This is an option requirement for independent work in topics or problems - refer to 1986 Catalog, page 31.

CSD MAJOR REQUIREMENTS

The following CSD major requirements are divided into three parts as follows:

Credits Requirement - the credit total

Course Requirement - courses total

Course Set courses taken in a required course set

refer to 1986 Catalog, page 66.

Credit Requirements:

The student has completed the following:

1. _____ of 45 required credits in CSD, and no more than 70.
2. _____ of 8 required credits in Human Anatomy and Physiology.
3. _____ of 5 required credits in Child Development.
4. _____ of 3 required credits in Statistics.
5. _____ of 5 required credits in Physiology/Neurosciences (this item is required beginning with the 1986 catalog).
6. _____ of 5 required credits in Philosophy (this item is required under the 1982 catalog).

Course Requirements

The student has yet to complete the following:

1. Required CSD courses:

to be filled by the courses which are not yet taken by the student and are required by the CSD Department from 1982-1986's catalogs, refer to 1986 catalog, page 66.

2. Required 400 level or above courses:

to be filled by the courses which are not yet taken by the student and are required by the CSD Department from 1982-1986's catalogs, refer to 1986 catalog, page 66.

3. PHYS 111 (yes/no), or _____ been exempted on the basis of 1 semester of Physics and 1 semester of Chemistry in high school).

Competency Requirements:

The student has achieved competency in the following areas:

1. Oral communication by completing: (usually satisfied by COMM 111, DRAMA 121, or extracurricular activities; minimum competence at advisor's discretion)
2. Written communication by completing: (usually satisfied by ENG102 or 110; minimum competence at advisor's discretion).

GER REQUIREMENTS

To receive a baccalaureate degree all students must complete, in addition to any other requirements, the following General Education Requirements (GER), refer to 1986 catalog, page 34.

student	enrolled	choice	catalog
non-transfer	84 autumn or after	must	NEW GER
transfer	86 autumn or after	must	NEW GER
all	84 autumn or before	may	OLD GER or NEW GER
transfer	86 autumn or before	may	OLD GER or NEW GER

Advisor notes:

The following requirements are for the student who is eligible to use OLD GER requirements.

OLD GER REQUIREMENTS

1. _____ credits of a required 12 in Behavioral and Social Sciences.
2. _____ credits of a required 12 in Humanities and the Arts.
3. _____ credits of a required 12 in Sciences and Mathematics.

NEW GER REQUIREMENTS

The following requirements differ according to the number of transfer credits held by the student.

The following groups will be used to identify applicable requirements.

Any student with 0 .. 40 transfer credits belongs to group 1

Any student with 41 .. 90 transfer credits belongs to group 2

Any student with over 91 transfer credits belongs to group 3

NEW GER - Competency Requirements

Requirements not identified by a group number are requirements for all students.

1. Writing Skill Requirement:

- a. ____ (taken/not) English 110, or ____ (been/not) exempted by a Writing Placement Exam (1).
- b. ____ of 1 required W prefix, 300 level or above.
- c. ____ of 3 required W prefix courses (1).
- d. ____ of 2 required W prefix courses (2).

2. Math Requirement:

- a. ____ (taken/not) Math course numbered 104 or above, or ____ (been/not) exempted by the Math Placement Test.

3. Foreign Language or Symbolic System Requirement:

Foreign Language:

- a. ____ (taken/not) the 3rd quarter of a FLL course sequence, (GER113, or RS123 are also included) or ____ (been/not) exempted by testing of the Center for Student Development or the Foreign Language and Literature Department.

OR

Symbolic System:

b. completed the following parts of the course sequences listed (any one sequence is required)

CS	101, 102, 103
LING	301, 311, 312
MATH	104, 105, 106
MATH	107, 108, 109
MATH	131, 132, 133
MATH	151, 152, 153
MATH	107, 108, 241
MATH	121, 241
MATH	121; PHAR 359, 362, 443, 463, 532
MUS	111, 112, 113, 137, 138, 139
PHIL	110, 111

or _____ (been/not been) exempted from the sequence by a placement test of the department(s) concerned.

NEW GER - Distribution Requirements

The following requirements are for group 1 and 2:

Group 2 only needs to complete 4 out of 6 of the following items, plus one course from the remaining 2 items:

1. _____ credits of required 6 in Expressive Arts.
2. _____ courses of a required 2 in Literary and Artistic Studies.

3. _____ courses of a required 3 in Historical and Cultural Studies, including
 - a. ___ course(s) of a required 1 from Western Historical and Cultural Studies,
 - b. ___ course(s) of a required 1 from Non-Western Historical and Cultural Studies.
4. _____ courses of a required 2 from Social Sciences
5. Ethical and Human Values Studies:
 - a. ___ courses of a required 1 from Group 1 of Ethical and Human Values.
 - b. ___ courses of a required 1 from Group 2 of Ethical and Human Values.
6. _____ courses of a required 3 in Natural Sciences, including
 - a. ___ courses of a required 1 with a lab in Natural Sciences.

The following requirements are for group 3:

1. _____ course of a required 1 in Expressive Arts
2. _____ of a required 1 course in Literary and Artistic Studies
3. _____ of a required 1 course in Historical and Cultural Studies
4. _____ of a required 1 course in Social Sciences
5. _____ of a required 1 course in Ethical and Human Values Studies
6. _____ of a required 1 course in Natural Sciences

NEW GER - Capstone Requirement

The student _____ (has/not) completed 1 required capstone course in the senior year.

Appendix D

Final Transformed Requirements

Instructions for reading **Appendix D**:

1. Read the ASRRB.DBF from top to bottom, left to right.
2. Column titles are displayed as original function-names.
3. Following are brief descriptions of the column titles, for detail descriptions please refer to the technical manual under the ASRRB.DBF section.

1. **Title:**

requirement title

2. **Item:**

requirement's item number

3. **ReqNum:**

number of credit or courses required

4. **UM:**

a flag indicating courses taken either as UM or Non-um options

5. **EX:**

a flag indicating courses taken either as Extension or Non-extension options

6. **OM:**

a flag indicating courses taken either as Omnibus or Non-Omnibus options

7. SmType:

the credit or course total in a course set

8. Oin:

a flag indicating that "in" is to be displayed before course file's name

9. CD:

code (number) of a department

10. OD:

a flag to display a department's abbreviation or whole name

11. GL:

an instruction to match courses' numbers

12. CN:

a course's number

13. CL:

a course's letter

14. Op:

an instruction to match files

15. CF:

the name of the file which is to be matched

16. CS:

the name of a course set to be matched

17. OSS:

an instruction to output a file's contents

18. OSTP:

an instruction to output files in different style

19. GER:

a flag to match student's **General Education Requirement** type

20. Exam:

the name of an examination or exemption to be checked

21. OE:

a flag to output the examination's name in normal or special style

22. Stype:

a flag to match the student's type

23. Sprule:

Special rule, an instruction to produce a requirement not covered under any other column.

24. AP1:

Extra message to be displayed

25. AP2:

Extra message to be displayed

26. AP3:

Extra message to be displayed

Appendix D

(ASRRB.DBF)

5

1	2	3	4	5	6	7	8	9	10	11	12	13
Title	Item	ReqNum	UM	EX	OM	SmType	OIn	CD	OD	GL	CN	CL
1	BAR	1	195			1	T	0	F		0	
2		2	93			1	T	0	F		0	
3		3	45	1	2	1	T	0	F		0	
4		4	30	1		1	T	0	F		0	
5		5	30	1	2	1	T	0	F		0	
6	UWER	1	0			3	T	0	F		0	
7	UMR	1	24			1	T	0	F		0	
8		2	12	1		1	T	0	F		0	
9	OR	1	40		1	1	T	0	F		0	
10	MRC	1	45			1	T	180	F		0	
11		2	8			1	T	0	F		0	
12		3	5			1	T	0	F		0	
13		4	3			1	T	0	F		0	
14		5	5			1	T	0	F		0	
15		6	5			1	T	147	T		0	
16	MCS	1	0				T	0	F		0	
17		2	14				T	180	F	>=	400	
18		3	0			4	F	153	F	=	111	
19	MCP	1	0				T	0	T		0	
20		2	0				T	0	F		0	
21	OGER	1	12			1	T	0	F		0	h
22		2	12			1	T	0	F		0	b
23		3	12			1	T	0	F		0	s
24	GCW		0				F	0	F		0	
25	GCW	1.a	0			2	F	190	F	=	110	
26		1.b	1			4	F	0	F	>=	300	W
27		1.c	3			4	F	0	F		0	W
28		1.d	2			4	F	0	F		0	W
29		2	0			1	F	141	F	>=	104	
30		3.a	0				T	0	F		0	
31		3.b	0				T	0	F		0	
32	GD	1	6			1	T	0	F		0	A
33		2	2			2	T	0	F		0	L
34		3	3			2	T	0	F		0	H
35		3.a	1			2	T	0	F		0	
36		3.b	1			2	T	0	F		0	
37		4	2			2	T	0	F		0	S
38		5.a	1			2	T	0	F		0	

Appendix D

(ASRRB.DBF)

6

1	2	3	4	5	6	7	8	9	10	11	12	13
Title	Item	ReqNum	UM	EX	OM	SmType	Oin	CD	OD	GL	CN	CL
39	5.b	1				2	T	0	F		0	
40	6	3				2	T	0	F		0	N
41	6.a	1				2	T	0	F		0	
42	GD 1	1				2	T	0	F		0	A
43	2	1				2	T	0	F		0	L
44	3	1				2	T	0	F		0	H
45	4	1				2	T	0	F		0	S
46	5	1				2	T	0	F		0	E
47	6	1				2	T	0	F		0	N
48	GC 1	1				4	F	0	F		0	

Appendix D

(ASRRB.DBF)

14	15	16	17	18	19	20	21	22	23
Op	CF	CS	OSS	OSTP	GER	Exam	OE	Stype	Sprule
1		BACS		0					
2	ACCAS	BACS		0					
3		BACS		0					
4		LBAC		0					
5		BACS		0					
6				0		UMWE	2		UWER
7		MNCS		0					
8		MNCS		0					
9		BACS		0					
10				0					
11	ACHAP			0					
12	ACCD			0					
13	ACST			0					
14	ACPN			0					
15				0					
16	1 ACCSDMR		A1	1					
17	2 ACCSD4		A2	1					
18				0		PHYCH	1		
19	ACDC		A3	2					
20	ACWC		A4	2					
21		NMCS		0	0				
22		NMCS		0	0				
23		NMCS		0	0				
24				0	N			23	
25				0	N	WPE	1	1	
26				0	N				
27				0	N			1	
28				0	N			2	
29				0	N	MPT	1		
30	ACFL			0	N	CSDFL	1		
31			A5	3	N				SS
32				0	N			12	
33				0	N			12	
34				0	N			12	
35	ACHCSW			0	N			12	
36	ACHCSNW			0	N			12	
37				0	N			12	

Appendix D

(ASRRB.DBF)

14 Op	15 CF	16 CS	17 OSS	18 OSTP	19 GER	20 Exam	21 OE	22 Stype	23 Sprule
38	ACEHV1			0	N			12	
39	ACEHV2			0	N			12	
40				0	N			12	
41				0	N			12	NSL

42				0	N			3	
43				0	N			3	
44				0	N			3	
45				0	N			3	
46				0	N			3	
47				0	N			3	

48	ACC			0	N				

24-AP1, 25-AP2, 26-AP3 are displayed if they are not empty

- 4 AP1:among the last 45 credits earned. This criterion
AP2: applies to graduating seniors only; National Exch
AP3:ange Students may be exempted (see catalog)
- 5 AP1:(for a second BA degree only)
- 6 AP1:credits must still be earned prior to eligibility
AP2: for the University Writing Examination
- 7 AP1:in an approved minor department have been earned
- 8 AP1:in an approved minor department have been earned
- 9 AP1:with an Omnibus course option for a B.A. have been
AP2: earned for independent work. Note: A maximum of
AP3: 15 per topic is allowed
- 10 AP1:., but no more than 70
- 14 AP1:(this item is required begining with the 1986 cata
AP2:log)
- 15 AP1:(this item is required under the 1982 catalog)
- 16 AP1:The student has not taken:
- 17 AP1:., have yet to be completed. This deficiency may be
AP2: filled by the following courses:
- 19 AP1:(usually satisfied by COMM 111, DRAMA 121, or extr
AP2:al curricular activities; minimum competence at ad
AP3:visor's discretion):
- 20 AP1:(usually satisfied by ENG 102 or 110; minimum comp
AP2:etence at advisor's discretion)
- 30 AP1:(or GER 113 or RS 123)
- 31 AP1:The student has yet to complete any one of the foi
AP2:lowing sequences of courses in symbolic systems by
AP3: taking the courses without an asterisk
- 41 AP1:with a NATURAL SCIENCE lab

Appendix E

System Configuration

Hardware Configuration:

The AUAS system requires the following hardware to function properly:

- IBM PC AT or compatible with at least 640K RAM
- IBM graphics printer or compatible
- (Color) Monitor
- One Hard Disk

Disk Space Requirements

The following table summarizes the amount of disk space each AUAS component requires.

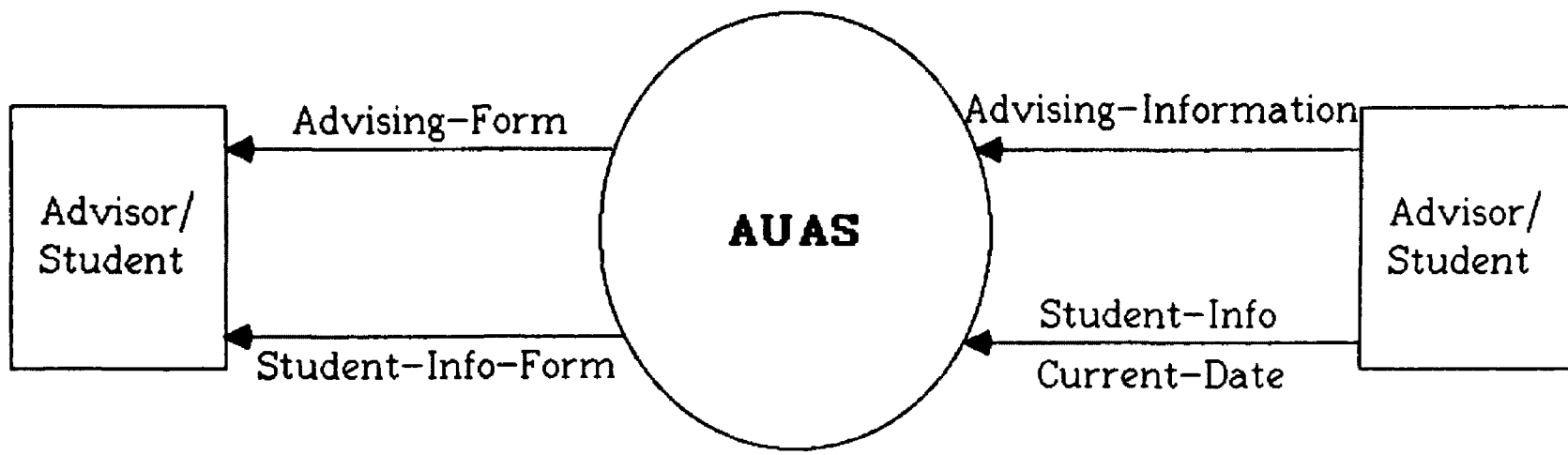
Component	Requires disk space	Fixed Size?
150 Programs	600K	yes
34 AUAS data bases	320K	yes
10 batch files	10K	yes
Student Course Data Bases	average 10K/per student who takes about 50 courses	no, depends on the # of courses taken

Notice that the Student Course Data Bases have variable sizes. It is the programmer's responsibility to ensure that there is enough disk space to store the student's data bases, AUAS does not check for disk full error.

Appendix F

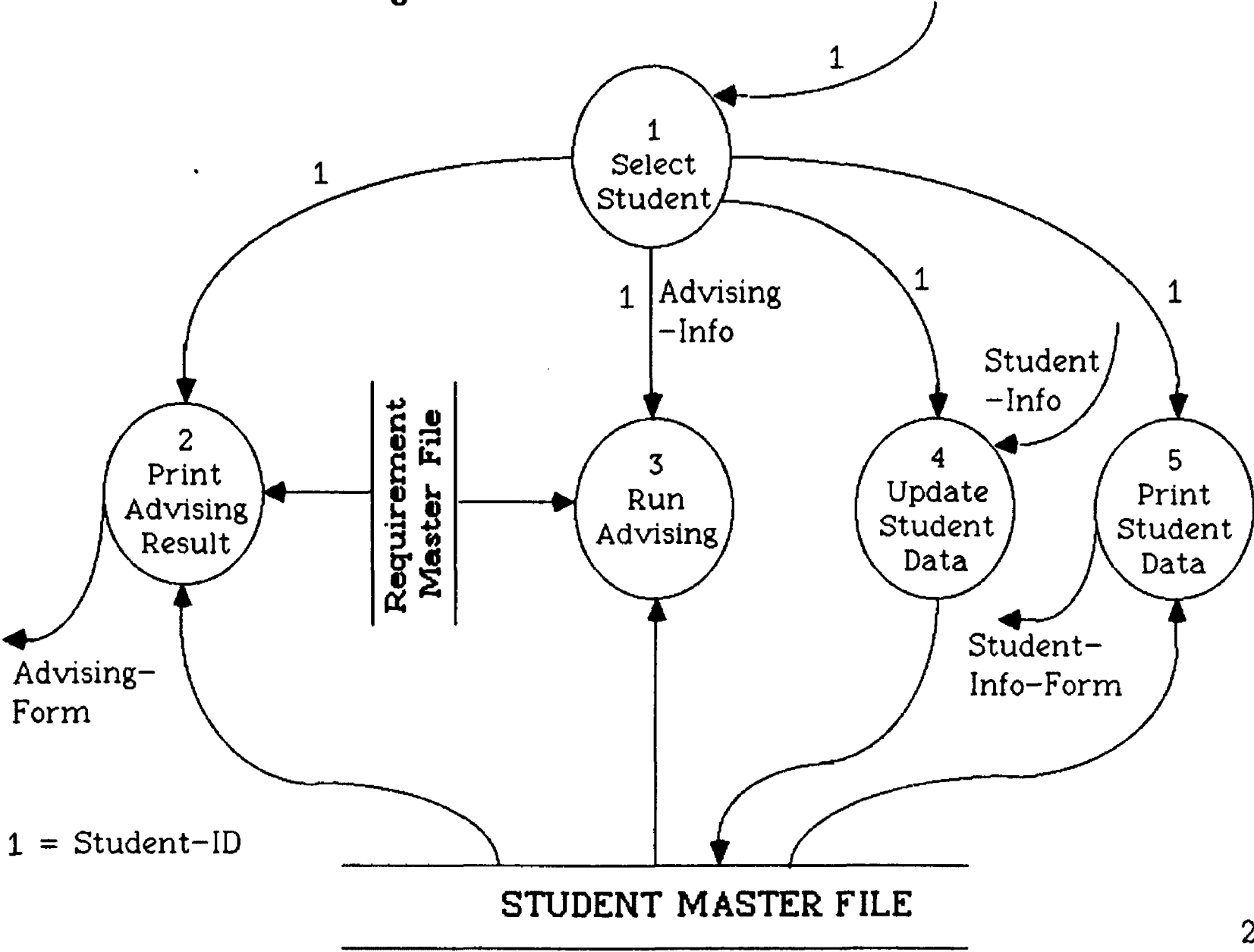
Data Flow Diagram

CONTEXT DIAGRAM



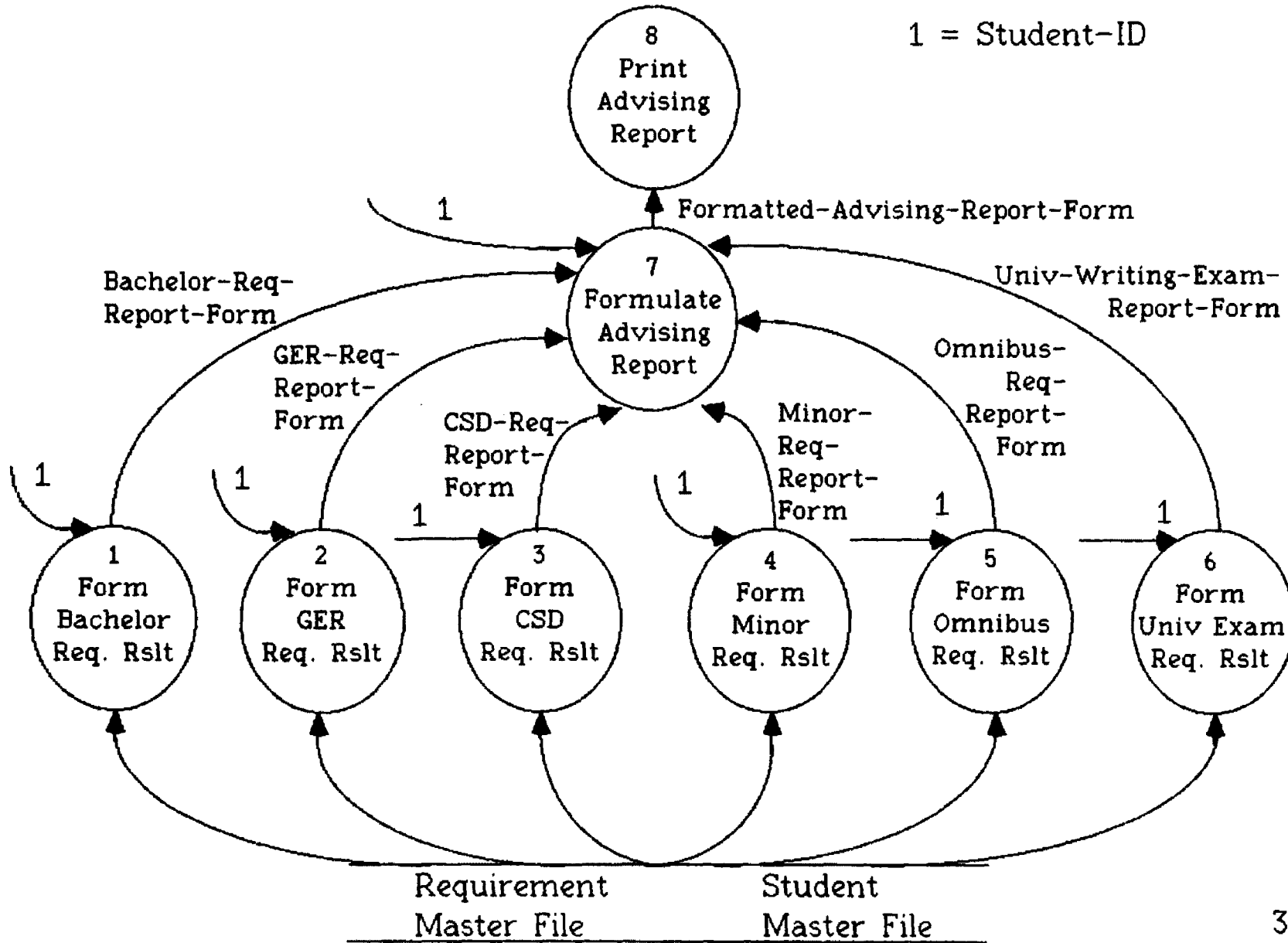
Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Level 0 Data Flow Diagram

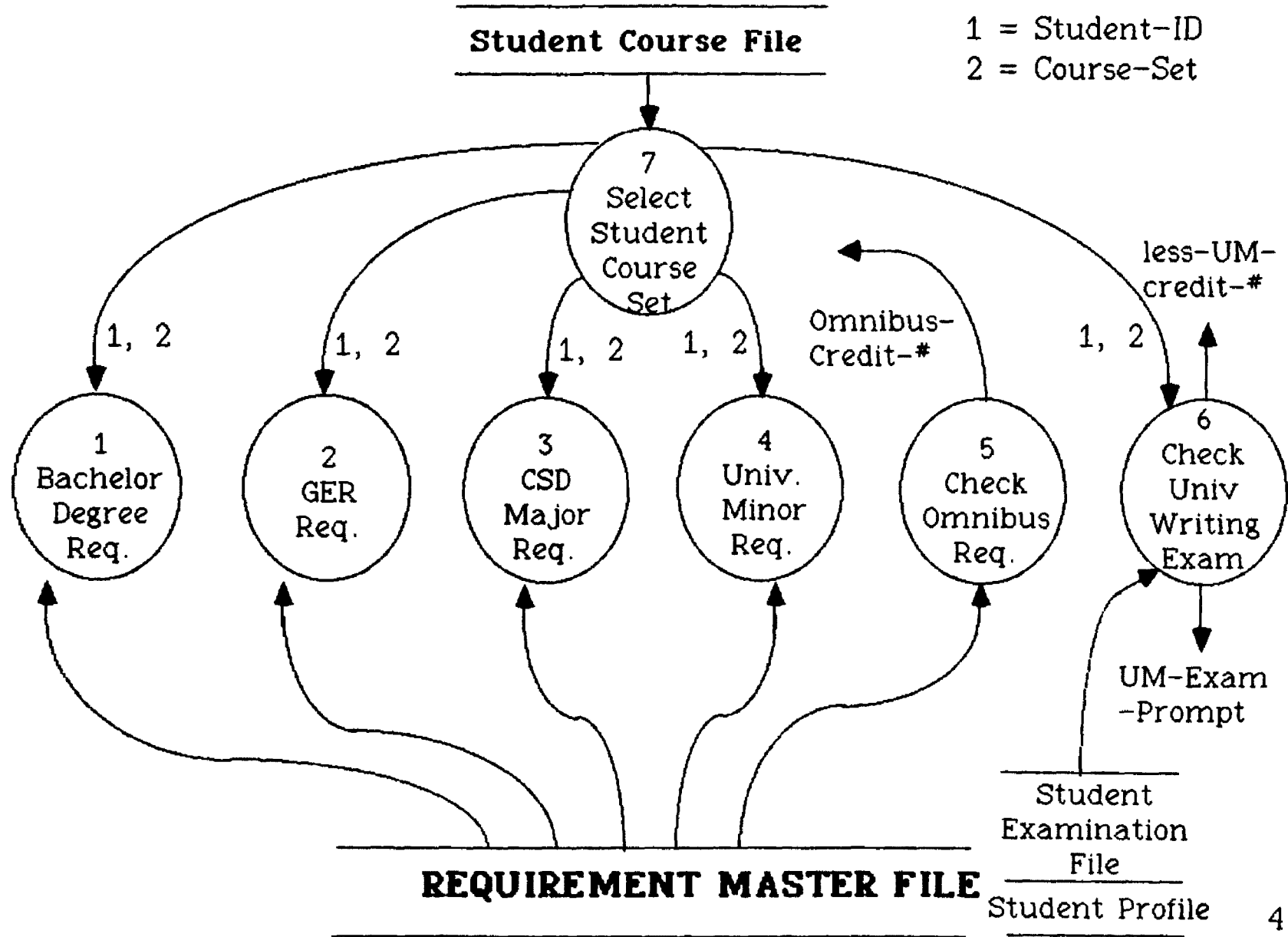


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

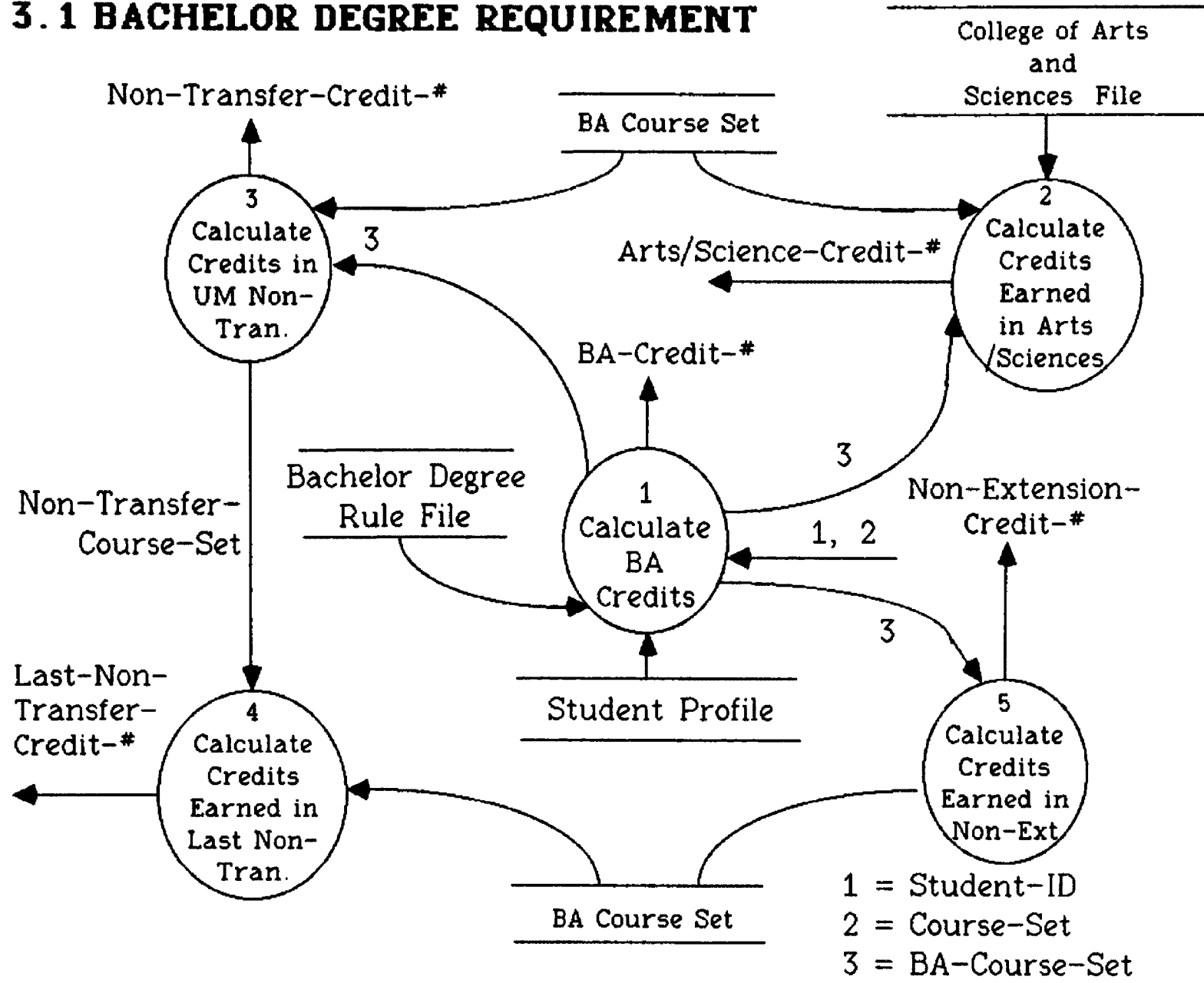
2.0 PRINT ADVISING RESULT



3.0 RUN ADVISING

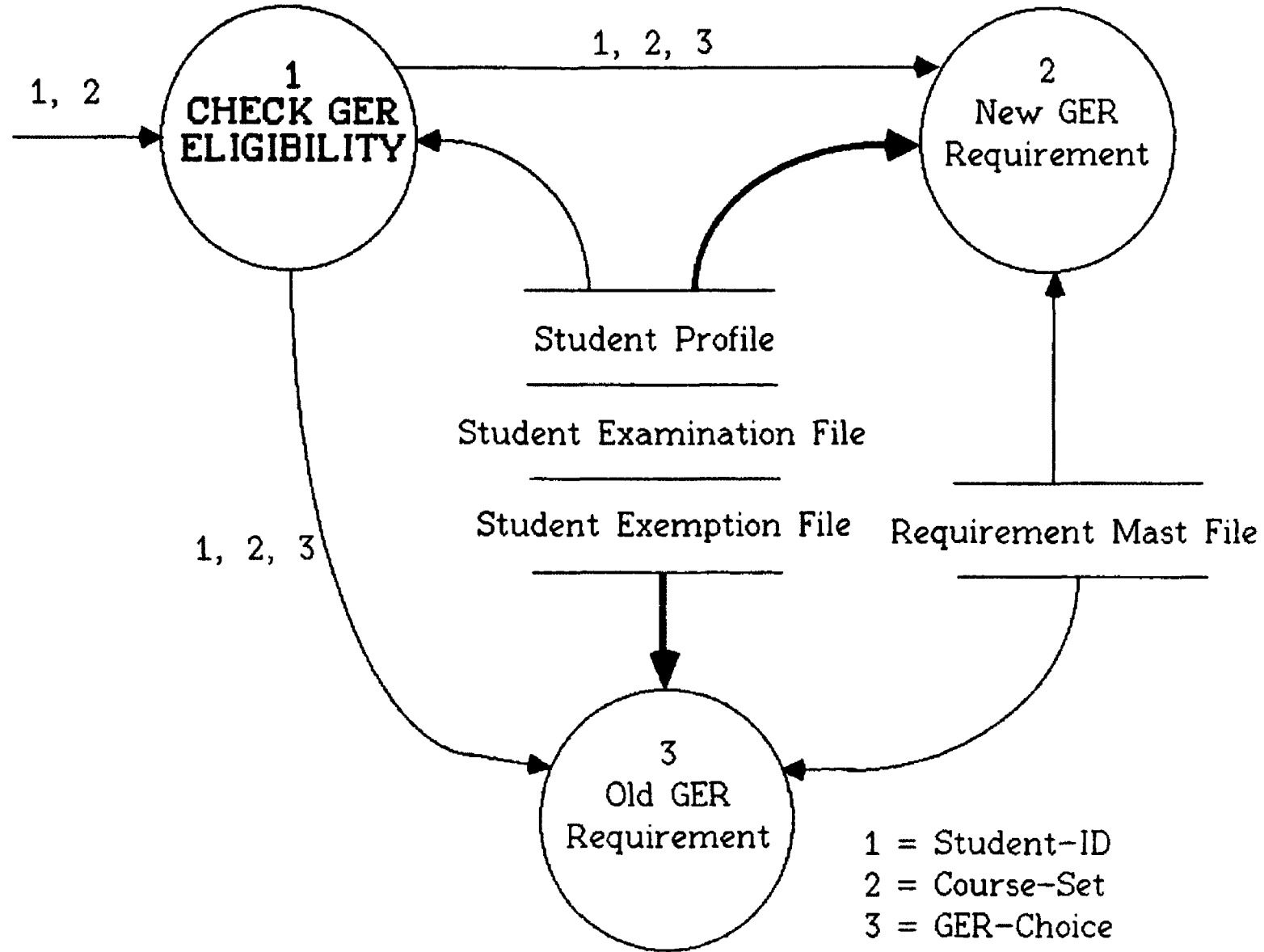


3. 1 BACHELOR DEGREE REQUIREMENT

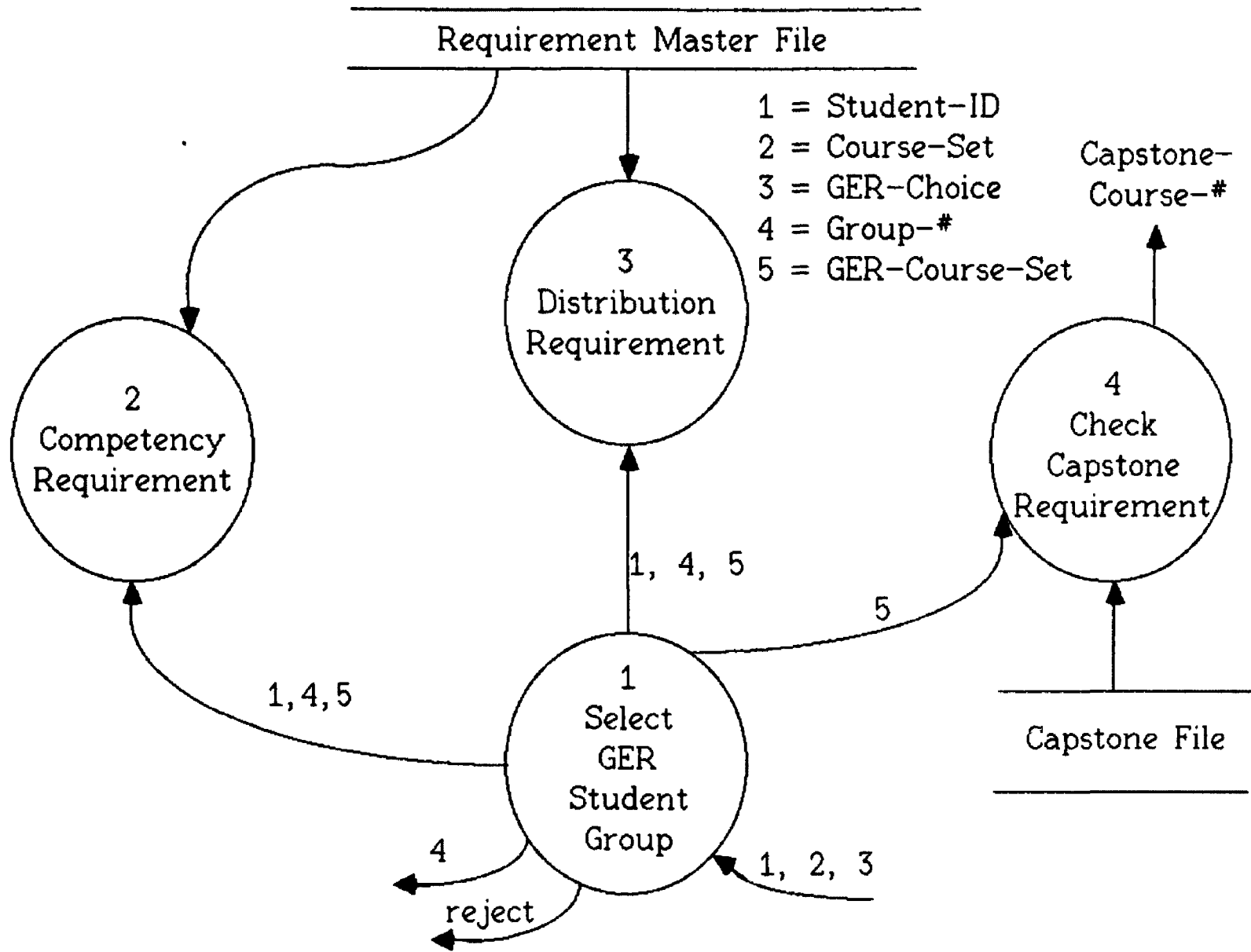


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

3.2 GER REQUIREMENT

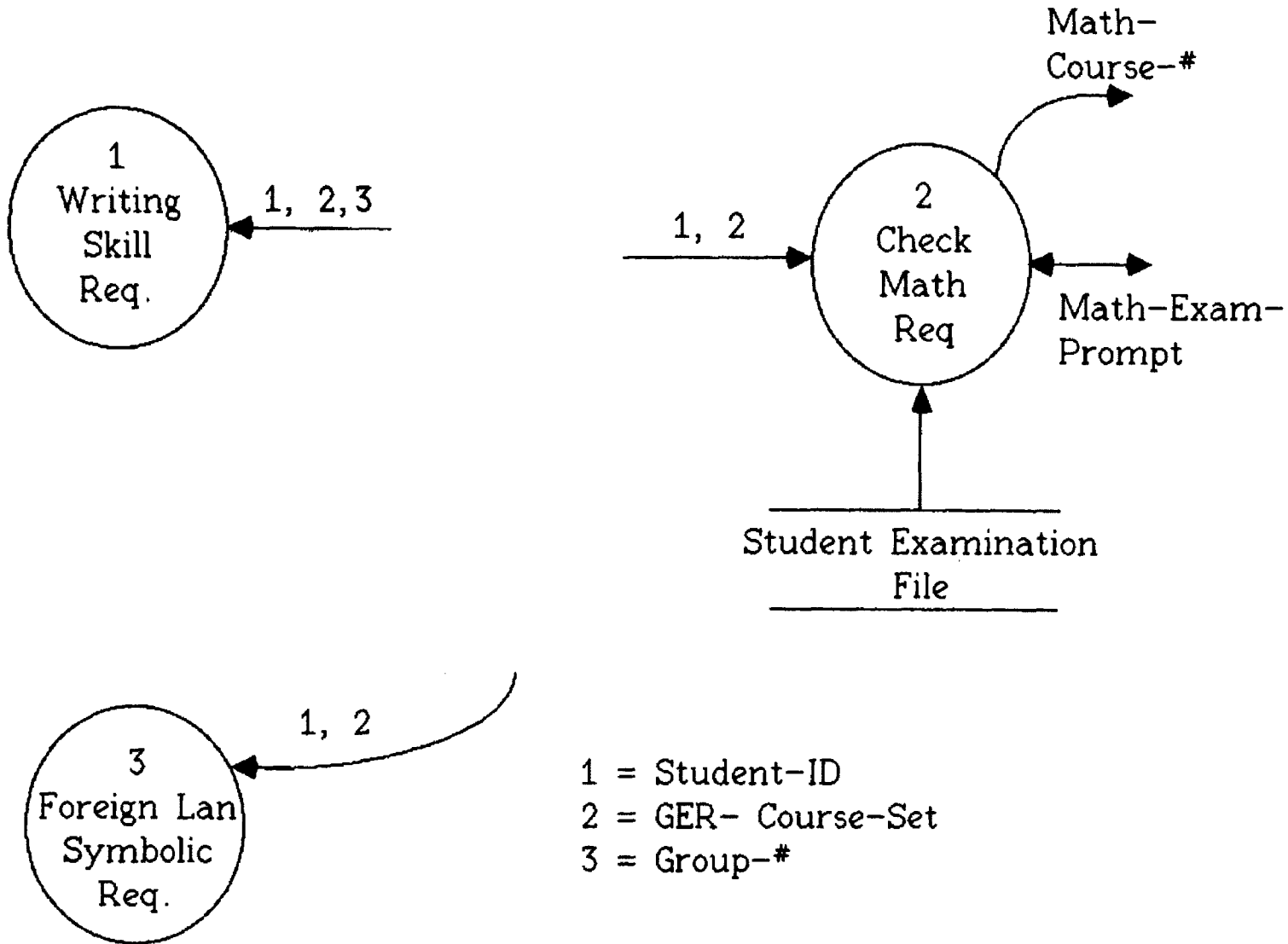


3.2.2 NEW GER REQUIREMENT

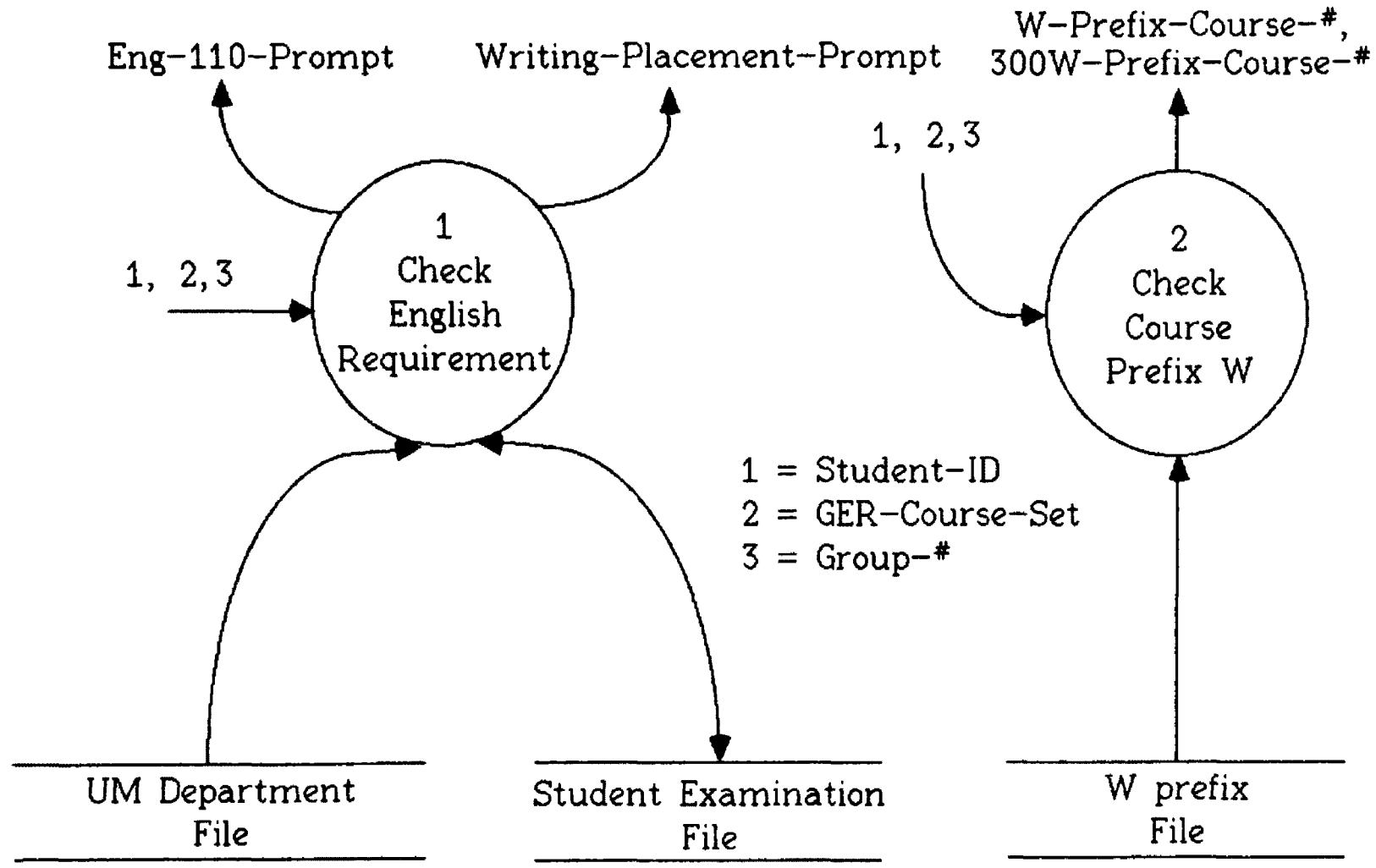


Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

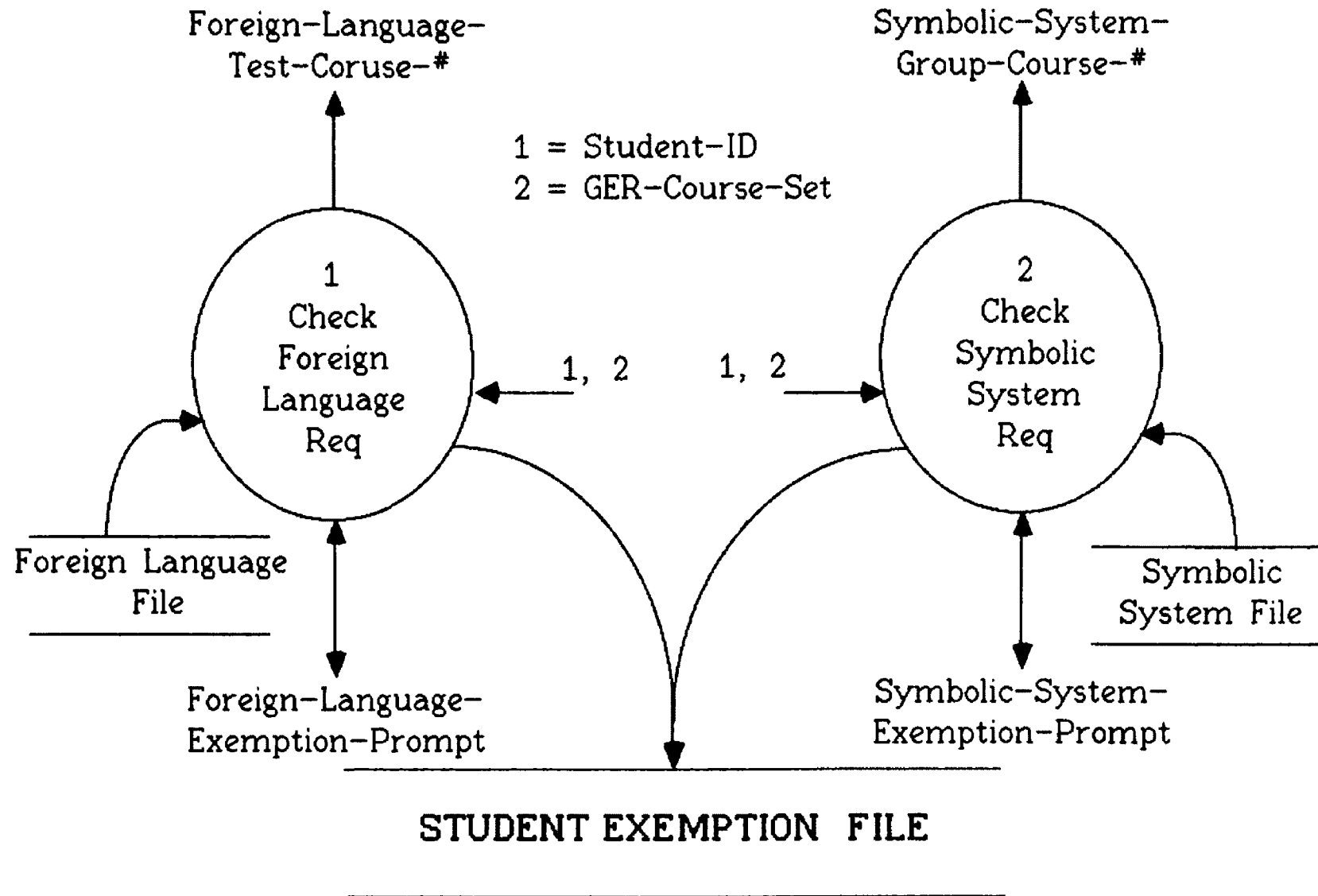
3.2.2.2 COMPETENCY REQUIREMENT



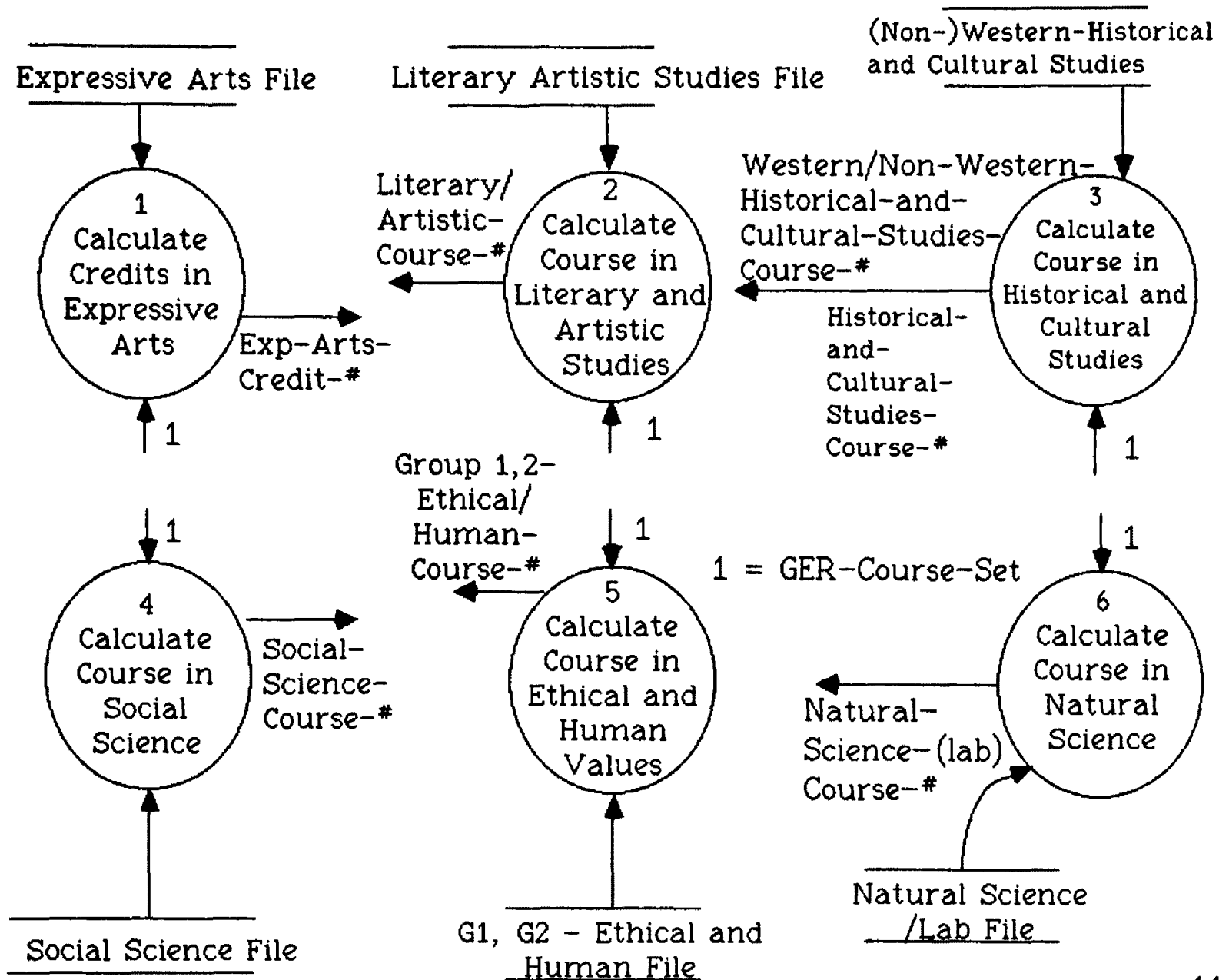
3.2.2.2.1 WRITING SKILL REQUIREMENT



3.2.2.2.3 Foreign Language/Symbolic System Requirement

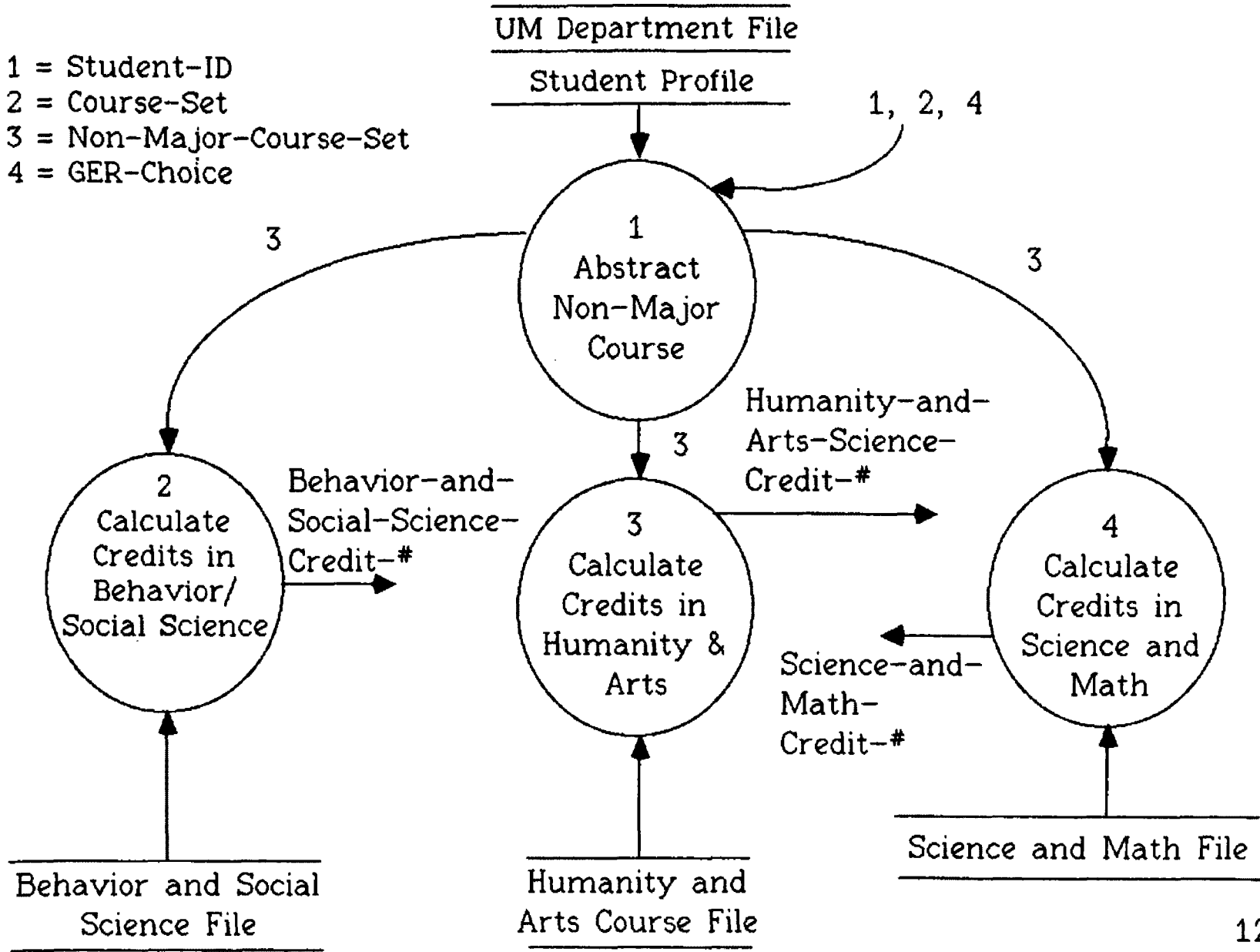


3.2.2.3 DISTRIBUTION REQUIREMENT

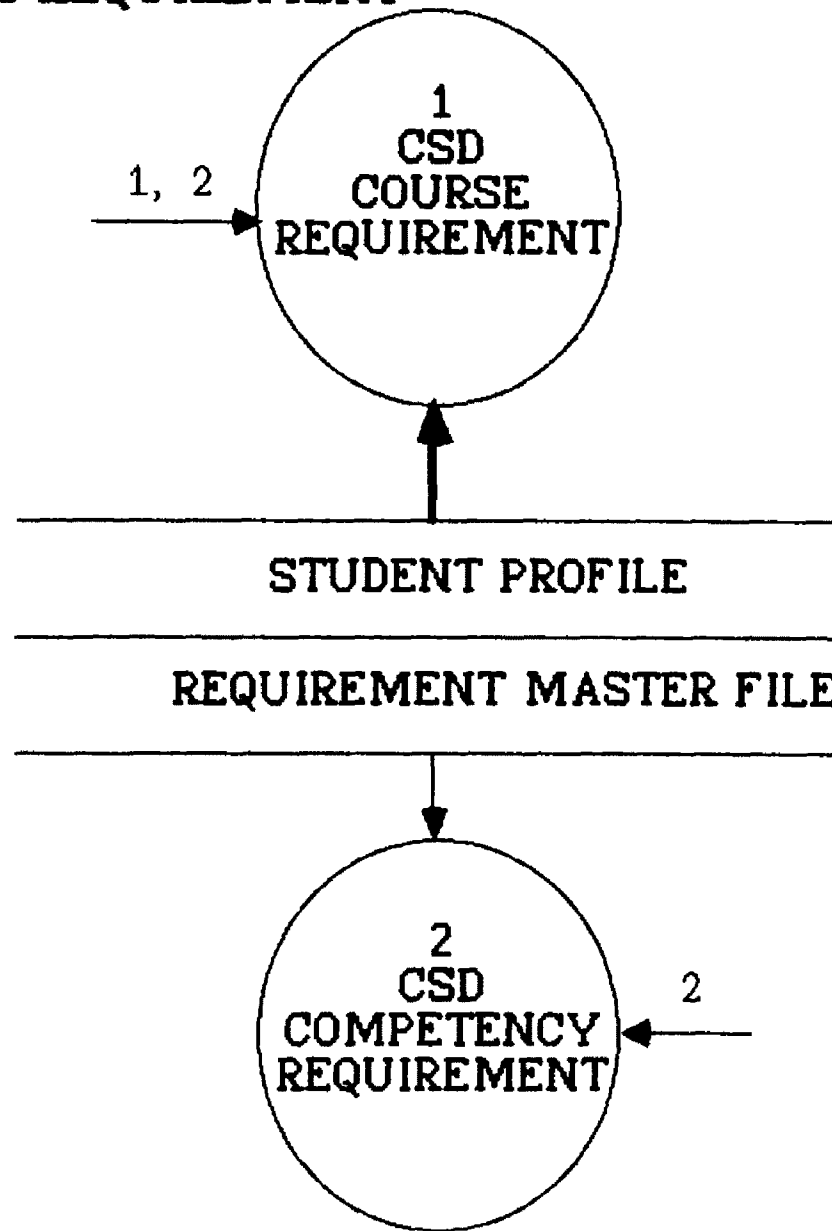


3.2.3 OLD GER REQUIREMENT

- 1 = Student-ID
- 2 = Course-Set
- 3 = Non-Major-Course-Set
- 4 = GER-Choice

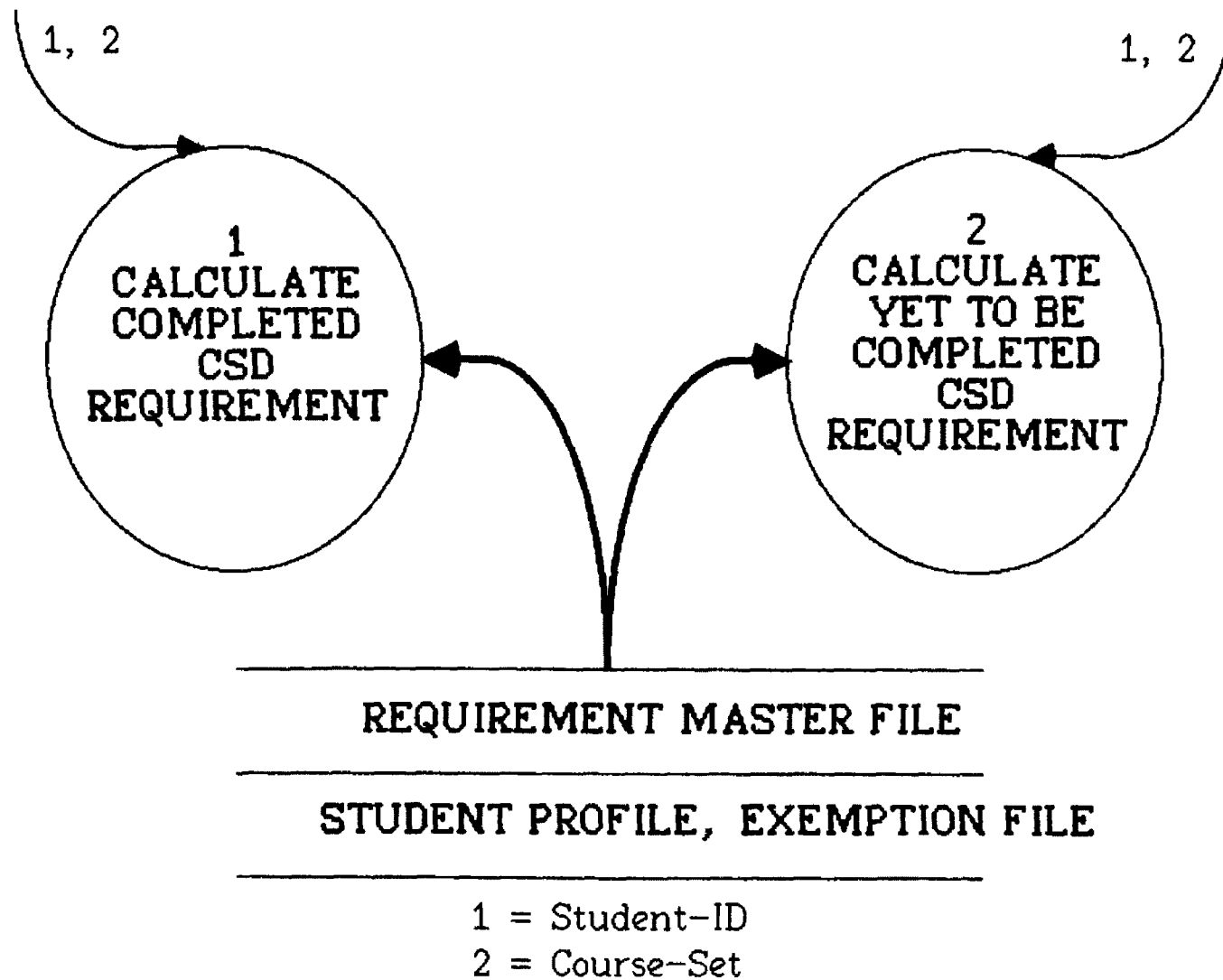


3.3 CSD MAJOR REQUIREMENT

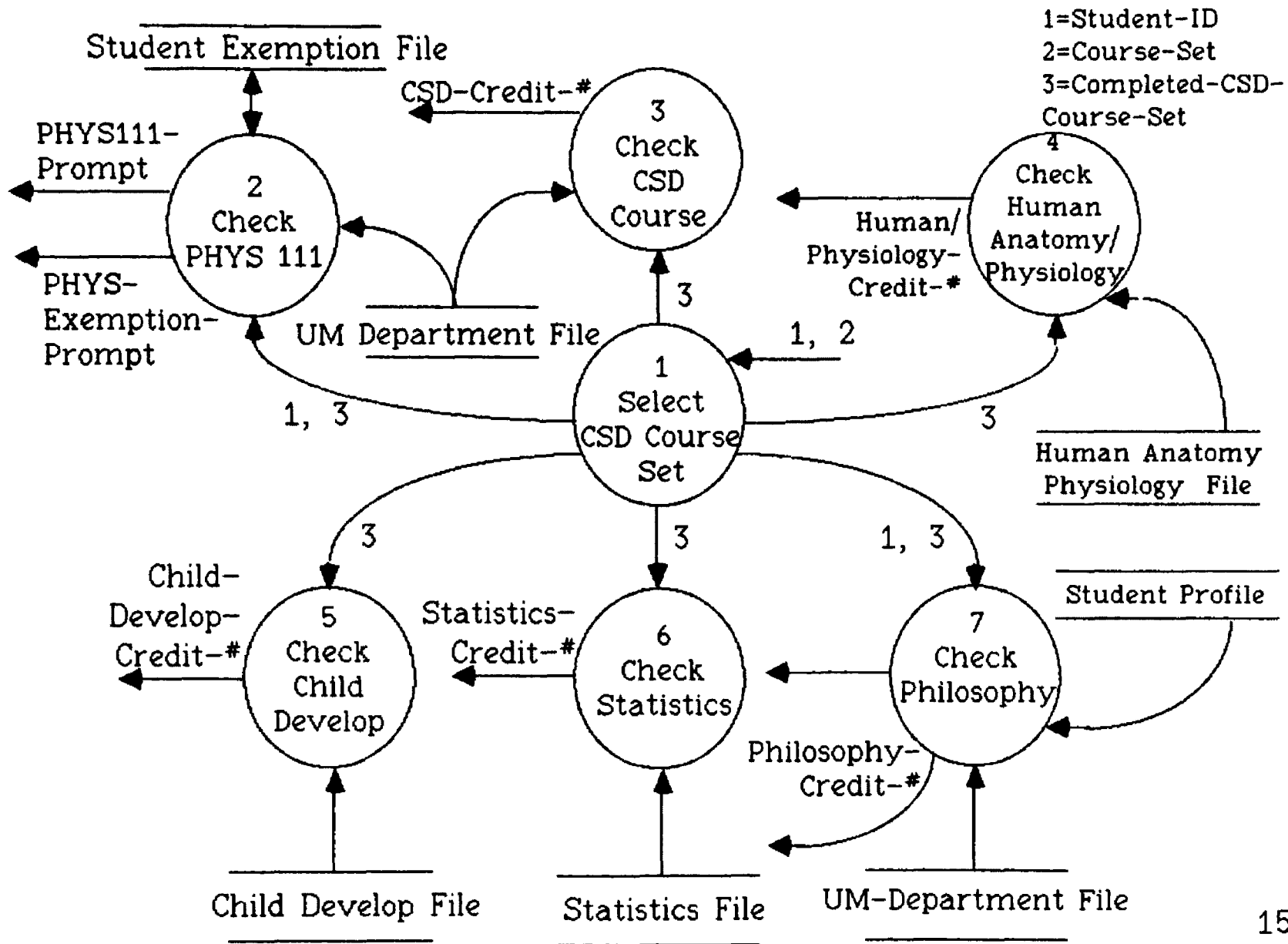


1 = Student-ID
2 = Course-Set

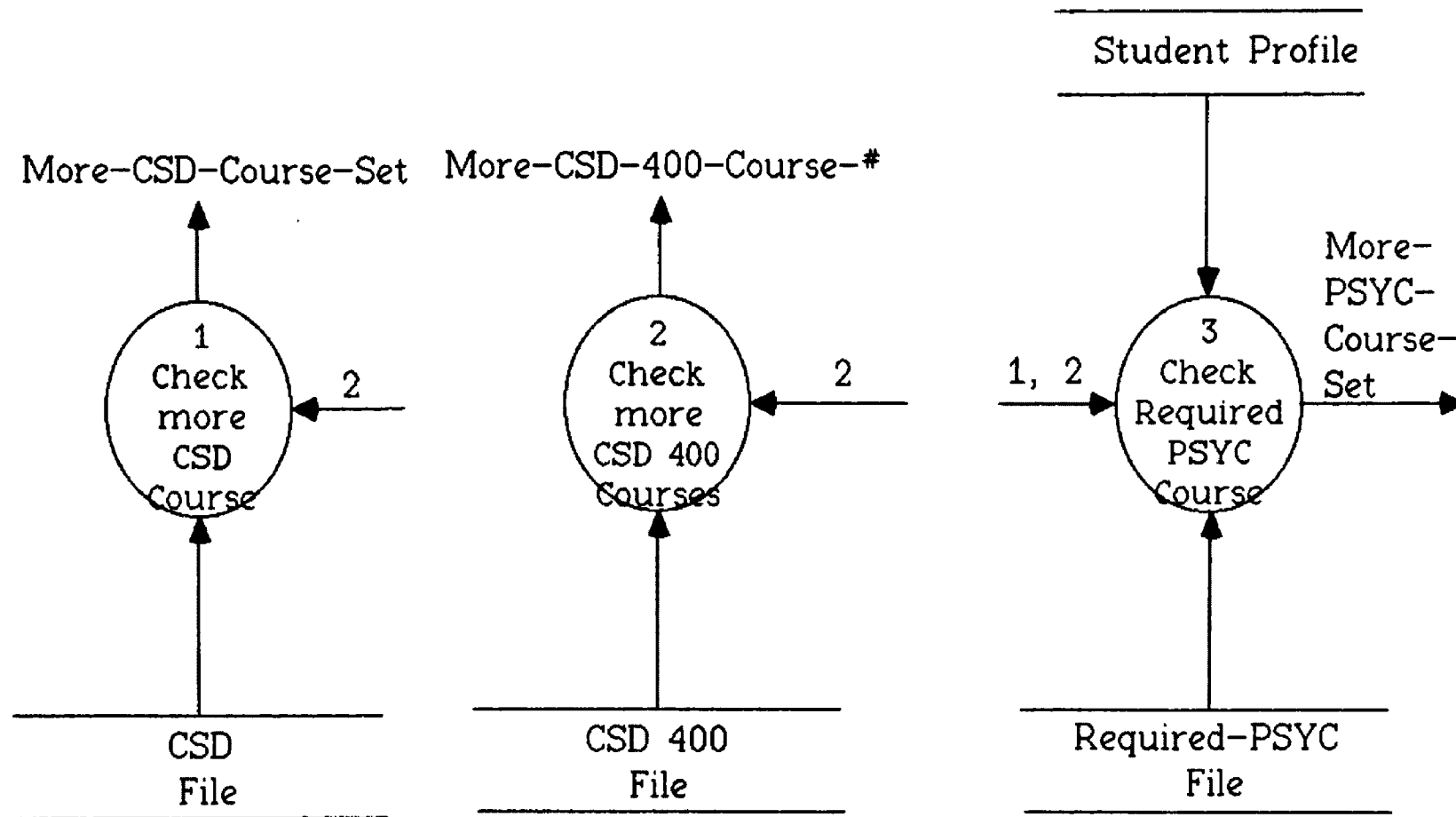
3.3.1 CSD MAJOR COURSES REQUIREMENT



3.3.1.1 CALCULATE COMPLETED CSD REQUIREMENT

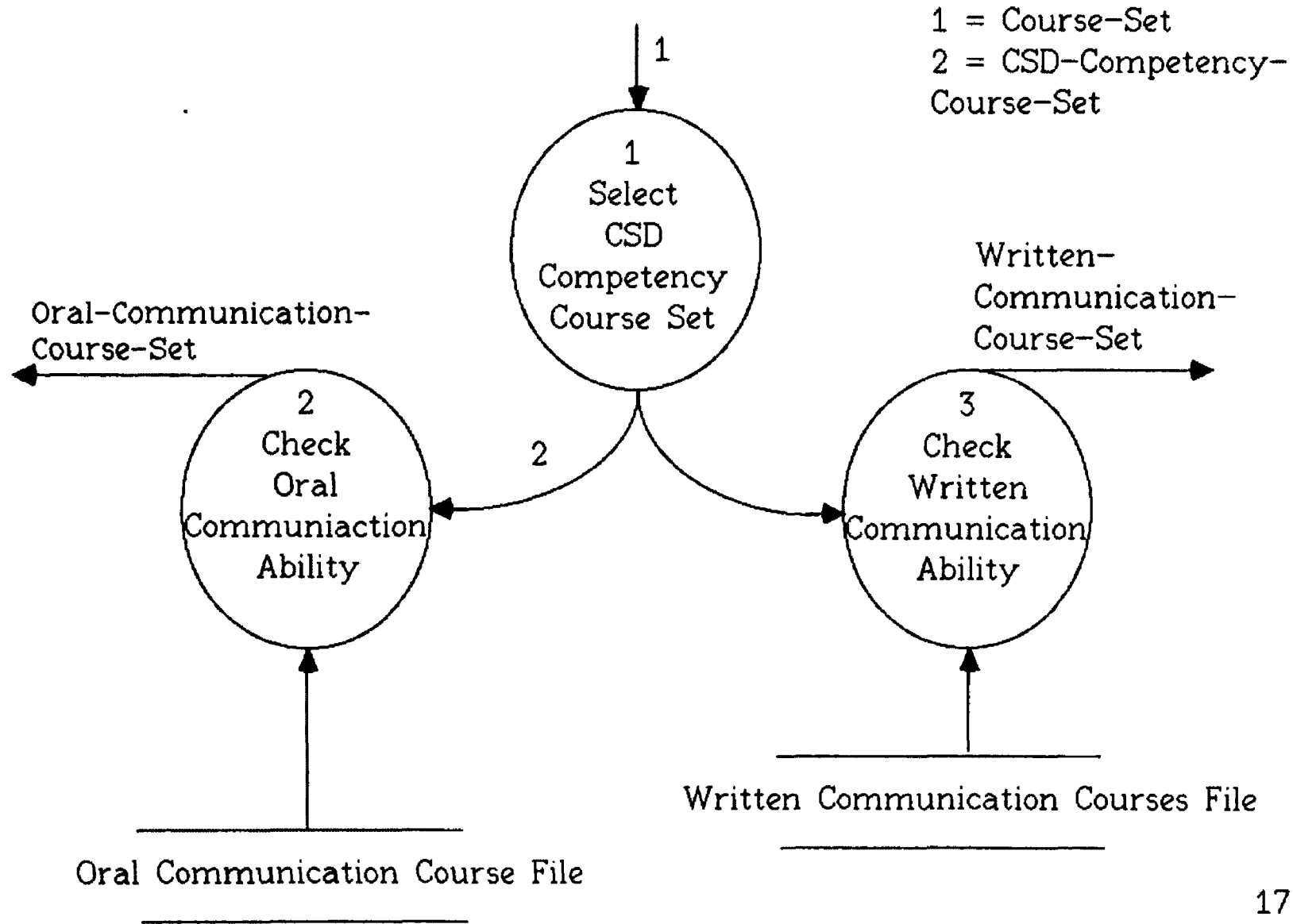


3.3.1.2 CALCULATE YET TO BE COMPLETED CSD REQUIREMENT

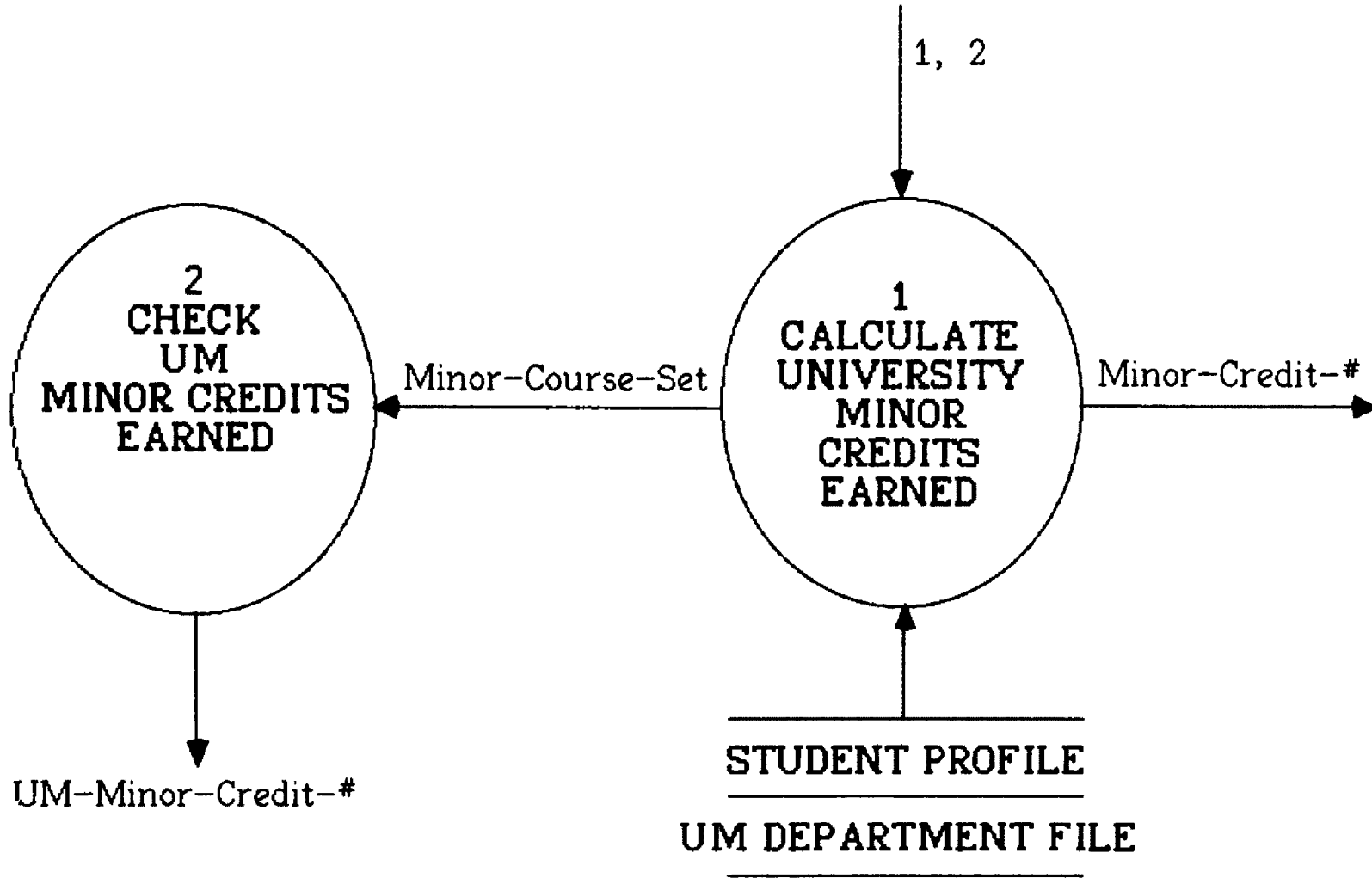


1 = Student-ID
2 = Course-Set

3.3.2 CSD COMPETENCY REQUIREMENT

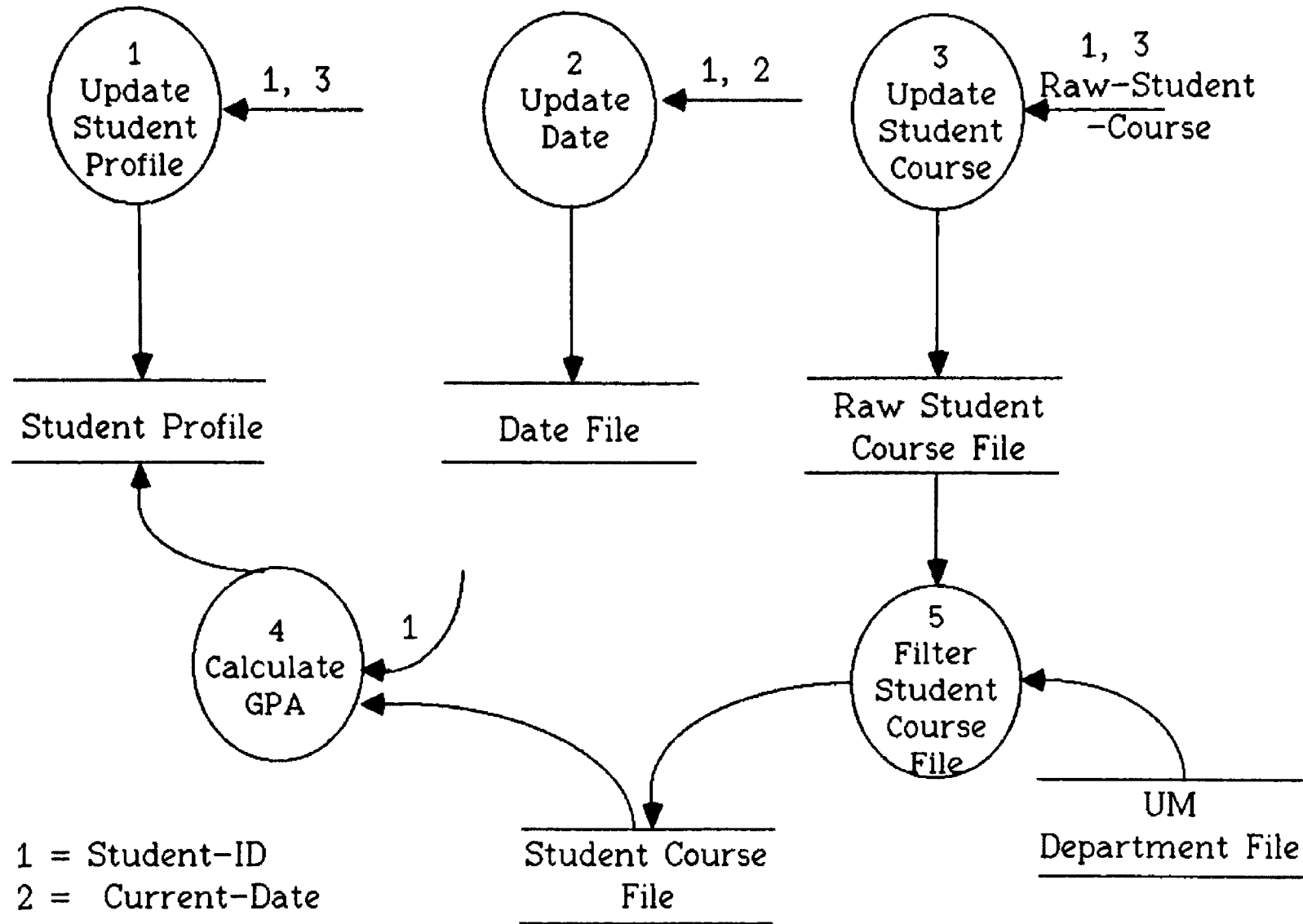


3.4 UNIVERSITY MINOR REQUIREMENT



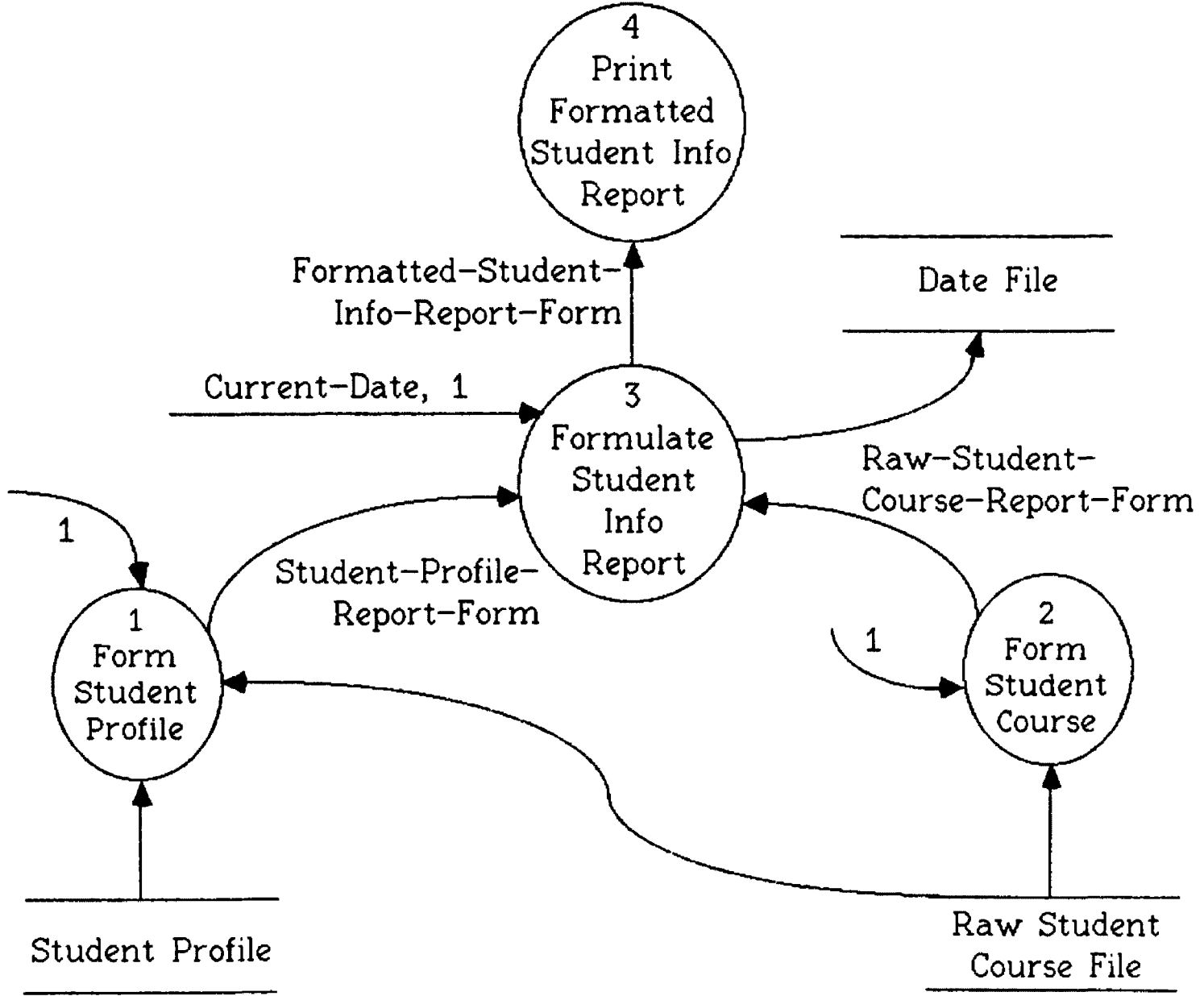
1 = Student-ID
2 = Course-Set

4.0 UPDATE STUDENT DATA



5.0 PRINT STUDENT DATA

1 = Student-ID



Reproduced with permission of the copyright owner. Further reproduction prohibited without permission.

Appendix G

Data Dictionary

.....

Data Flow Name : Advising-Form

Alias :

Composition :

A printout for all corresponding Advising Process Result.

Note : For an equivalent form, please reference to Appendix E.

.....

.....

Data Flow Name : Advising-Information

Alias :

Composition :

A set of all advising-related input/output prompts of the AUAS.

.....

.....

Data Flow Name : BA-Course-Set

Alias :

Composition :

A set of the following :

- (Course-ID +
- Credits +
- Quarter-Offered +
- Year-Offered +
- Extension +
- UM-Course)

.....

.....

Data Flow Name : Completed-CSD-Course-Set

Alias :

Composition :

A set of the following :

- { Course-ID +
- Credits +
- Quarter-Offered +
- Year-Offered +
- Grade }

.....

.....

Data Flow Name : Course-ID

Alias :

Composition :

- Department-Code +
- Course-Number

.....

.....

Data Flow Name : Course-Set

Alias :

Composition :

A set of the following data flow :

- { Student-Course }

.....

.....

Data Flow Name : CSD-Competency-Course-Set

Alias :

Composition :

A set of the following :
{ Course-ID +
Credits }

.....

.....

Data Flow Name : Formatted-Advising-Report-Form

Alias :

Composition :

A set of all related Advising report forms, which is formatted for output to printer.

.....

.....

Data Flow Name : Formatted-Student-Info-Report-Form

Alias :

Composition :

A set of all related Student Data report forms, which is formatted for output to printer.

.....

.....

Data Flow Name : GER-Course-Set

Alias :

Composition :

A set of the following :

- { Course-ID +
- Credits +
- Quarter-Offered +
- Year-Offered +
- Grade +
- UM-Course }

.....

.....

Data Flow Name : Historical-and-Cultural-Studies-Course-Set

Alias :

Composition :

A set of the following :

- { Course-ID +
- Year-Offered +
- Quarter-Offered +
- Credits +
- Grade }

.....

.....

Data Flow Name : Minor-Course-Set

Alias :

Composition :

A set of the following :

- { Credits +
- UM-Course }

.....

.....

Data Flow Name : More-CSD-Course-Set

Alias :

Composition :

A set of the following :

{ Course-ID +
Credits +
Course-Title }

.....

.....

Data Flow Name : More-PSYC-Course-Set

Alias :

Composition :

A set of the following :

{ Course-ID +
Credits +
Course-Title }

.....

.....

Data Flow Name : Non-Major-Course-Set

Alias :

Composition :

A set of the following :

{ Course-ID +
Credits +
Quarter-Offered +
Year-Offered }

.....

.....

Data Flow Name : Non-Transfer-Course-Set

Alias :

Composition :

A set of the following :

{ Credits +
Quarter-Offered +
Year-Offered }

.....

.....

Data Flow Name : Oral-Communication-Course-Set

Alias :

Composition :

A set of the following :

{ Course-ID +
Credits +
Course-Title +
Grade }

.....

.....

Data Flow Name : Raw-Student-Course

Alias :

Composition :

- Department +
- Course-# +
- Course-Title +
- Year-Offered +
- Quarter-Offered +
- Credits +
- Grade +
- Omnibus +
- Non-Degree +
- Extension +
- Repeat +
- UM-Course

.....

.....

Data Flow Name : Raw-Student-Course-Set

Alias :

Composition :

A set of Raw-Student-Course

.....

.....

Data Flow Name : Required-PSYC-Course-Set

Alias :

Composition :

- o(Course-ID +
- Course-Title +
- Credits +
- Grade)

.....

.....

Data Flow Name : Student-Course

Alias :

Composition :

- (Course-ID +
- Course-Title +
- Year-Offered +
- Quarter-Offered +
- Credits +
- Grade +
- Omnibus +
- Non-Degree +
- Extension +
- Repeat +
- UM-Course)

.....

.....

Data Flow Name : Student-Info

Alias :

Composition :

A set of all student-related information of the AUAS.

.....

.....

Data Flow Name : Student-Info-Form

Alias :

Composition :

A printout for all corresponding Student information.

.....

.....

Data Flow Name : Student-Profile-Data

Alias :

Composition :

- Student-ID +
- Student-Name +
- Major-Department +
- Minor-Department +
- GER-Year-Start +
- GER-Year-End +
- Major-Year-Start +
- Major-Year-End +
- Transfer-Status +
- Year-Enrolled

.....

.....

Data Flow Name : Student-Profile-Set

Alias :

Composition :

A set of Student-Profile-Data.

.....

.....

Data Flow Name : Written-Communication-Course-Set

Alias :

Composition :

- { Course-ID +
- Credits +
- Course-Title +
- Grade }.

.....

.....

Data Flow Name : X-Report-Form

Alias :

Composition :

X is composed by the following components.

- Bachelor-Req,
- GER-Req,
- CSD-Req,
- Minor-Req,
- Omnibus-Req,
- Student-Profile,
- Raw-Student-Course,
- Univ-Writing-Exam-Req.

Which is a set of all corresponding (X) data information, ready for formatting.

.....

6.2 Data Element Dictionary

.....

Data Element Name : Course-Title

Alias :

Value :

10 character string, which represents UM course title.

Notes : Example *CS 542* has the course title "Design"

.....

.....

Data Element Name : Credits

Alias :

Value :

2 digits number, value 1 .. 15

.....

.....

Data Element Name : Current-Date

Alias :

Value :

mm/dd/yy

mm : Month value 1 .. 12

dd : Day value 1 .. 31

yy : Year value 00 .. 99

.....

.....

Data Element Name : Department

Alias :

Value :

4 character string, abbreviation for UM department.

Notes : Example Math for Mathematical department

.....

.....

Data Element Name : Department-Code

Alias :

Value :

3 digits number, which represents UM department.

Notes : Example 112 for computer science department

.....

.....

Data Element Name : Extension

Alias :

Value :

Logical Variable, value true/false (T/F).

.....

.....

Data Element Name : GER-Choice

Alias :

Value :

Value "New" or "All"

.....

.....

Data Element Name : GER-Year-End

Alias :

Value :

2 digits number, value 00 .. 99

.....

.....

Data Element Name : GER-Year-Start

Alias :

Value :

2 digits number, value 00 .. 99

.....

.....

Data Element Name : Grade

Alias :

Value :

1 character string, value A, B, C, D, F, I, N, P, X, W

.....

.

.....

Data Element Name : Group-#

Alias : Group-Number

Value :

Number value 1, 2, 3

1 : Group 1

2 : Group 2

3 : Group 3

.....

.....

Data Element Name : Major-Year-Start

Alias :

Value :

2 digits number, value 00 .. 99

.....

.....

Data Element Name : Minor-Year-Start

Alias :

Value :

2 digits number, value 00 .. 99

.....

.....

Data Element Name : Minor-Department

Alias :

Value :

Four character string. Abbreviation for UM's Department.

.....

.....
Data Element Name : Major-Department
Alias :
Value :

Four character string. Abbreviation for UM's Department.

.....
Data Element Name : Non-Degree
Alias :
Value :

Logical Variable, value true/false (T/F).

.....
Data Element Name : Omnibus
Alias :
Value :

Logical Variable, value true/false (T/F).

.....
Data Element Name : Quarter-Enrolled
Alias :
Value :

- 1 character string.
- A : Autumn
- W : Winter
- S : Spring
- U : Summer
- O : Other

.....

.....

Data Element Name : Quarter-Offered

Alias :

Value :

1 character string.

A : Autumn

W : Winter

S : Spring

U : Summer

O : Other

.....

.....

Data Element Name : Repeat

Alias :

Value :

Logical Variable, value true/false (T/F).

.....

.....

Data Element Name : Student-ID

Alias :

Value :

###-##-####

: 0 .. 9

.....

.....

Data Element Name : Transfer-Status

Alias :

Value :

Value : "non-Transfer" or "Transfer"

.....

.....

Data Element Name : UM-Course

Alias :

Value :

Logical Variable, value true/false (T/F).

.....

.....

Data Element Name : Year-Enrolled

Alias :

Value :

##

Value 00 .. 99

.....



Data Element Name : X-Course-#

Alias :

Value :

Numeric Number value 0 .. 99.

X is composed by the following components :

- Foreign-Language-Test,
- Group1-Ethical/Human,
- Group2-Ethical/Human,
- Historical-and-Cultural-Studies,
- Literary/Artistic,
- Math,
- More-CSD-400,
- Natural-Science,
- Natural-Science-Lab,
- Non-Western-Historical-and-Cultural-Studies,
- Social-Science,
- Symbolic-System-Group-#,
- W-Prefix,
- Western-Historical-and-Cultural-Studies,
- 300W-Prefix.





Data Element Name : X-Credit-#

Alias : X-Credit-Number

Value :

Numerical Digits value 0 .. 299

X is composed by the following components

- Arts/Science,
- BA,
- Behavior-and-Social-Science,
- Capstone,
- Child-Develop,
- CSD,
- CSD-400-Level,
- Exp-Arts,
- Human/Physiology,
- Humanity-and-Arts-Science,
- Less-UM,
- Last-Non-Transfer,
- Minor,
- Non-Transfer,
- Non-Extension,
- Omnibus,
- Philosophy,
- Science-and-Math,
- Social-Science,
- Statistics,
- Transfer,
- UM,
- UM-Minor



.....

Data Element Name : X-Prompt

Alias :

Value :

A mark - X, one character string.

X is composed by the following components :

- Foreign-Language-Exemption,
- Math-Exam,
- Eng-110,
- PHYS-111,
- PHYS-Exemption,
- Symbolic-System-Exemption,
- UM-Exam,
- Writing-Placement.

.....

.....

Data Element Name : Year-Enrolled

Alias :

Value :

2 digits number, value 00 .. 99

.....

.....

Data Element Name : Year-Offered

Alias :

Value :

2 digits number, which represents the time of the year of a UM course offered.

Notes : Example CS 542 is offered in 1986, Winter, then
the Year-Offered - 86.

.....

Appendix H

Process Description

.....

Process Number : 1.0
Process Name : Select-Student
Access File : None
Process Description :

Repeat
 Prompt for Student-ID
 confirm Student-ID
Until no Student-ID is entered.

.....

Process Number : 2.1
Process Name : Form-Bachelor-Requirement-Result
Access File : Requirement-Master-File,
 Student-Master-File.
Process Description :

Run through Advising program "batch" (without interacting with user), save screen output to a file image.

.....

Process Number : 2.2
Process Name : Form-GER-Requirement-Result
Access File : Requirement-Master-File,
 Student-Master-File.
Process Description :

Run through Advising program "batch" (without interacting with user), save screen output to a file image.

.....

Process Number : 2.3
Process Name : Form-CSD-Requirement-Result
Access File : Requirement-Master-File,
 Student-Master-File.
Process Description :

Run through Advising program "batch" (without interacting with user), save screen output to a file image.

.....

Process Number : 2.4
Process Name : Form-Minor-Requirement-Result
Access File : Requirement-Master-File,
Student-Master-File.

Process Description :

Run through Advising program "batch" (without interacting with user), save screen output to a file image.

.....

Process Number : 2.5
Process Name : Form-Omnibus-Requirement-Result
Access File : Requirement-Master-File,
Student-Master-File.

Process Description :

Run through Advising program "batch" (without interacting with user), save screen output to a file image.

.....

Process Number : 2.6
Process Name : Form-University-Writing-Examination-Requirement-Result
Access File : Requirement-Master-File,
Student-Master-File.

Process Description :

Run through Advising program "batch" (without interacting with user), save screen output to a file image.

.....

Process Number : 2.7
Process Name : Formulate-Advising-Report
Access File : None
Process Description :

For each advising-Report-Form
Provide heading information and corresponding report format.

.....

Process Number : 2.8
Process Name : Print-Advising-Report
Access File : None
Process Description :

Print Formatted-Advising-Report-Form.

.....

Process Number : 3.1.1
Process Name : Calculate-BA-Credits
Access File : Student-Profile,
Bachelor-Degree-Rule-File.
Process Description :

For each course in Course-Set
Check through each rule in **Bachelor-Degree-Rule-File**
against **with Student-Profile** by Student-ID
If no rule is violated
Then
Add Credits to BA-Credit-#.
Add corresponding data into BACS.

.....

Process Number : 3.1.2
Process Name : Calculate-Credits-Earned-in-Arts-and-Sciences
Access File : College-of-Arts-and-Sciences-File
Process Description :

For each course in BA-Course-Set
Check if Department-Code is in **College-of-Arts-and-Sciences-File**
If yes then
Add Credits to Arts-and-Sciences-Credit-#.

.....

Process Number : 3.1.3
Process Name : Calculate-Credits-in-UM-Non-Transfer
Access File : None
Process Description :

For each course in BA-Course-Set
If UM-Course is "True"
Then
 Add Credits to Non-Transfer-Credit-#.
 Add corresponding Course data into Non-Transfer Course-Set.

.....

Process Number : 3.1.4
Process Name : Calculate-Credits-Earned-in-Last-Non-Transfer
Access File : None
Process Description :

Sort course in Non-Transfer-Course-Set by Year-Offered and Quarter-Offered. (by Descending Order)
Repeat
 Add Credits to Last-Non-Transfer-Credit-#.
Until Last-Non-Transfer-Credit-# >= 45 or End of Non-Transfer-Course-Set.

.....

Process Number : 3.1.5
Process Name : Calculate-Credits-Earned-in-Non-Extension
Access File : None
Process Description :

For each course in BA-Course-Set
If Non-Extension = "True"
Then
 Add Credits to Non-Extension-Credit-#.

.....

Process Number : 3.2.1

Process Name : Check-GER-Eligibility

Access File : Student-Profile

Process Description :

Find Transfer-Status, Year-Enrolled, Quarter-Enrolled in **Student-Profile** by Student-ID.

If Year-Enrolled, Quarter-Enrolled <= 84 Autumn

Then

GER-Choice = "All".

If Transfer-Status = "non-transfer"

Then

If Year-Enrolled, Quarter-Enrolled >= 84 Autumn

Then

GER-Choice = "New".

If Transfer-Status = "transfer"

Then

If Year-Enrolled, Quarter-Enrolled >= 86 Autumn

Then

GER-Choice = "New"

Else

GER-Choice = "All".

.....

Process Number : 3.2.2.1

Process Name : Select-GER-Student-Group

Access File : None

Process Description :

If GER-Choice - "All" or new

Then

For each course in Course-Set

Add course to GER-Course-Set.

If UM-Course - "False"

Then

Add credits to Transfer-Credits.

Case Transfer-Credits :

0 .. 40 : Set Group-# - 1

41 .. 90 : Set Group-# - 2

Over 91 : Set Group-# - 3

.....

Process Number : 3.2.2.2.1.1

Process Name : Check-English-Requirement

**Access File : UM-Department-File,
Student-Examination-File.**

Process Description :

If Student-Type <> 1 then quit

Else do the following

Check English-Department-Code from **UM-Department-File**.

For each course in GER-Course-Set

If Course-ID = English-Department-Code-110

Then

Eng-110-Prompt = "true"

Else

Eng-110-Prompt = "false".

Find Writing-Exam in **Student-Examination-File** by Student-ID.

If Writing-Exam <> "true"

Then

Prompt Writing-Placement-Prompt for new value if available.

.....

Process Number : 3.2.2.2.1.2

Process Name : Check-Course-Prefix-W

Access File : W-Prefix-File

Process Description :

For each course in GER-Course-Set

If course can be found in **W-Prefix-File**, and student-type = 1
or 2

Add 1 to W-Prefix-Course-#.

If Course-Number >= 300

Then

Add 1 to 300W-Prefix-Course-#.

.....

Process Number : 3.2.2.2.2
Process Name : Check-Math-Requirement
Access File : Student-Examination-File
Process Description :

Find Math-Department-Code in **UM-Department-File**.
For each course in **GER-Course-Set**
 If Department-Code = Math-Department-Code and
 Course-Number >= 104
 Then
 Add 1 to Math-Course-#.
Find Math-Exam in **Student-Examination-File** by Student-ID.
If found
Then
 Set Math-Exam-Prompt to be "True"
Else
 Prompt for new value if available.

.....

Process Number : 3.2.2.2.3.1
Process Name : Check-Foreign-Language-Requirement
Access File : Foreign-Language-File,
 Student-Exemption-File
Process Description :

For each course in GER-Course-Set
 If Course-ID can be found in **Foreign-Language-File**
 Then
 Add 1 to Foreign-Language-Test-Course-#.

If Foreign-Language-Exempt can be found in **Student-Exemption-File** by Student-ID
Then
 Foreign-Language-Exemption-Prompt = "true"
Else
 Prompt new value if available.

.....

Process Number : 3.2.2.2.3.2

Process Name : Check-Symbolic-System-Requirement

Access File : Student-Exemption-File,
Symbolic-System-File.

Process Description :

For each Symbolic-Group-# in **Symbolic-System-File**

If Course-ID can be found in GER-Course-Set

Then

Mark the Symbolic-Group-# in Symbolic-System-Group-Course-#.

Prompt Symbolic-system-Exemption-Prompt in **Student-Exemption-File** if the value <> 'true'.

.....

Process Number : 3.2.2.3.1

Process Name : Calculate-Credits-in-Expressive-Arts

Access File : Expressive-Arts-File

Process Description :

For each course in GER-Course-Set

If Course-ID can be found in **Expressive-Arts-File**

Then

Add Credits to Expressive-Arts-Credit-#.

.....

Process Number : 3.2.2.3.2

Process Name : Calculate-Course-in-Literary-and-Artistic-Studies

Access File : Literary-and-Artistic-Studies-File

Process Description :

For each course in GER-Course-Set

If Course can be found in **Literary-and-Artistic-Studies-File**

Then

Add 1 to Literary-and-Artistic-Studies-Course-#.

.....

Process Number : 3.2.2.3.3
Process Name : Calculate-Course-in-Historical-and-Cultural-Studies
Access File : Historical-and-Cultural-Studies-File
 Western-Historical-and-Cultural-Studies-File
 Non-Western-Historical-and-Cultural-Studies-File
Process Description :

For each course in GER-Course-Set
 If the course can be found in **Historical-and-Cultural-Studies-File**
 Then
 Add 1 to Historical-and-Cultural-Course-#.
 If the course can be found in **Western-Historical-and-Cultural-Studies-File**
 Then
 Add 1 to Western-Historical-and-Cultural-Course-#.
 If the course can be found in **Non-Western-Historical-and-Cultural-Studies-File**
 Then
 Add 1 to Non-Western-Historical-and-Cultural-Course-#.

.....

Process Number : 3.2.2.3.4
Process Name : Calculate-Course-in-Social-Science
Access File : Social-Science-File
Process Description :

For each course in GER-Course-Set
 If Course-ID can be found in **Social-Science-File**
 Then
 Add 1 to Social-Science-Course-#.

.....

Process Number : 3.2.2.3.5
Process Name : Calculate-Credits-in-Ethical-and-Human-Values
Access File : Group1-Ethical-and-Human-File
 Group2-Ethical-and-Human-File
Process Description :

For each course in GER-Course-Set
 If Course can be found in **G1-Ethical-and-Human-File**
 Then
 Add 1 to Group1-Ethical-and-Human-Course-#.
 If Course can be found in **G2-Ethical-and-Human-File**
 Then
 Add 1 to Group2-Ethical-and-Human-Course-#.

.....

Process Number : 3.2.2.3.6
Process Name : Calculate-Course-in-Natural-Science
Access File : Natural-Science-File
Process Description :

For each course in GER-Distribution-Course-Set
 If Course can be found in **Natural-Science-File**
 Then
 Add 1 to Natural-Science-Course-#.
 If Course can be found in **Natural-Science-Lab-File**
 Then
 Add 1 to Natural-Science-Lab-Course-#.

.....

Process Number : 3.2.2.4
Process Name : Check-Capstone-Requirement
Access File : Capstone-File
Process Description :

For each course in GER-Course-Set
 If course data can be found in **Capstone-File**.
 Then
 Add 1 to Capstone-Course-#.

.....

Process Number : 3.2.3.1
Process Name : Abstract-Non-Major-Course
Access File : Student-Profile,
UM-Department-File.

Process Description :

If GER-Choice - "All"
Then

Find Major-Department in Student-Profile by Student-ID.
Find Major-Department-Code in UM-Department-File
by Major-Department.

For each course in Course-Set

 If Department-Code <> Major-Department-Code
 Then

 Add corresponding Course data to Non-
 Major-Course-Set.

.....

Process Number : 3.2.3.2
Process Name : Calculate-Credits-in-Behavior-and-Social-Science
Access File : Behavior-and-Social-Science-File

Process Description :

For each course in Non-Major-Course-Set
 If course data can be found in **Behavior-and-Social-**
 Science-File
 Then

 Add Credits to Behavior-and-Social-Science-Credit-#.

.....

Process Number : 3.2.3.3
Process Name : Calculate-Credits-in-Humanity-and-Arts
Access File : Humanity-and-Arts-File

Process Description :

For each course in Non-Major-Course-Set
 If course data can be found in **Humanity-and-Arts-File**
 Then

 Add Credits to Behavior-and-Humanity-and-Arts-#.

.....

Process Number : 3.2.3.4
Process Name : Calculate-Credits-in-Science-and-Math
Access File : Science-and-Math-File
Process Description :

For each course in Non-Major-Course-Set
If course data can be found in **Science-and-Math-File**
Then
Add Credits to Science-Math-Credit-#.

.....

Process Number : 3.3.1.1.1
Process Name : Select-CSD-Course-Set
Access File : None
Process Description :

For each course in Course-Set
Dump Course-ID, Credits, Quarter-Offered, Year-Offered,
Grade to Completed-CSD-Course-Set.

.....

Process Number : 3.3.1.1.2
Process Name : Check-PHYS-111
Access File : Student-Exemption-File,
 UM-Department-File.

Process Description :

Find PHYS-code in **UM-Department-File**.

For each course in Completed-CSD-Course-Set

 If Course-ID = PHYS-code, 111

 Then

 Set PHYS-111-Prompt = 'True'

 Quit.

If PHYS-111-Prompt <> "True"

 Then

 Check PHYS-111-Exempt in **Student-Exemption-File** by
 Student-ID.

 If not found

 Then

 Prompt PHYS-Exemption-Prompt for new value if
 available.

.....

Process Number : 3.3.1.1.3
Process Name : Check-CSD-Course
Access File : UM-Department-File
Process Description :

Find CSD-code in **UM-Department-File**.

For each course in Completed-CSD-Course-Set

 If Department-Code = CSD-code

 Then

 Add Credits to CSD-Credits-#.

.....

Process Number : 3.3.1.1.4
Process Name : Check-Human-Anatomy-and-Physiology
Access File : Human-Anatomy-and-Physiology-File
Process Description :

For each course in Completed-CSD-Course-Set
If Course-ID can be found in **Human-Anatomy-and-Physiology-File**
Then
Add Credits to Human-Anatomy-and-Physiology-Credits
#.

.....

Process Number : 3.3.1.1.5
Process Name : Check-Child-Development
Access File : Child-Develop-File
Process Description :

For each course in Completed-CSD-Course-Set
If Course-ID can be found in **Child-Develop-File**
Then
Add Credits to Child-Develop-Credit-#.

.....

Process Number : 3.3.1.1.6
Process Name : Check-Statistics
Access File : Statistics-File
Process Description :

For each course in Completed-CSD-Course-Set
If Course-ID can be found in **Statistics-File**
Then
Add Credits to Statistics-Credit-#.

.....

Process Number : 3.3.1.1.7
Process Name : Check-Philosophy
Access File : Student-Profile,
UM-Department.

Process Description :

Find Major-Year-Start, Graduate-Year-End in **Student-Profile** by Student-ID.

If Major-Year-Start and Major-Year-End is between 82 and 83
Then

Find Philosophy-code in **UM-Department-File**.
For each course in Completed-CSD-Course-Set
If Department-Code - Philosophy-code
Then
Add Credits to Philosophy-Credits-#.

.....

Process Number : 3.3.1.2.1
Process Name : Check-Required-CSD-Course
Access File : CSD-File
Process Description :

For each course in **CSD-File**
If the course can not be found in Course-Set
Then
Dump Course-ID, Credits, Course-Title into More-CSD
Course-Set.

.....

Process Number : 3.3.1.2.2
Process Name : Calculate-400Level-Courses
Access File : CSD-400-File
Process Description :

For each course in **CSD-400-File**
If the course can not be found in Course-Set
Then
Dump Course-ID, Credits, Course-Title into More-400
Level-CSD-Course-Set.

.....

Process Number : 3.3.1.2.3
Process Name : Check-Required-PSYC-Course
Access File : Required-PSYC-File,
 Student-Profile.
Process Description :

For each course in **Required-PSYC-File**
If the course can not be found in Course-Set
Then
 Dump Course-ID, Credits, Course-Title into More-PSYC-Course-Set.

Find Major-Year-Start and Major-Year-End in **Student-Profile** by Student-ID.

If Major-Year-Start and Major-Year-End is between 82 and 83
Then
 If PSYC 111 or PSYC 300 is in More-PSYC-Course-#
 Then
 Delete the course from More-PSYC-Course-#.

.....

Process Number : 3.3.2.1
Process Name : Select-CSD-Competency-Course-Set
Access File : None
Process Description :

For each course in Course-Set
 Dump Course-ID, Credits, Grade into CSD-Competency-Course-Set.

.....

Process Number : 3.3.2.2
Process Name : Check-for-Oral-Communication-Ability
Access File : Oral-Communication-Course-File
Process Description :

For each course in CSD-Competency-Course-Set
If Course-ID can be found in **Oral-Communication-course-File**
Then
Add course to Oral-Communication-Course-Set.

.....

Process Number : 3.3.2.3
Process Name : Check-for-Written-Communication-Ability
Access File : Written-Communication-Course-File
Process Description :

For each course in CSD-Competency-Course-Set
If Course-ID can be found in **Written-Communication-Course-File**
Then
Add course to Written-Communication-Course-Set.

.....

Process Number : 3.4.1
Process Name : Calculate-University-Minor-Credits-Earned
Access File : Student-Profile,
UM-Department-File.
Process Description :

Find Minor-Department in **Student-Profile** by Student-ID.
Find Minor-Department-Code in **UM-Department-File** by the Minor-Department.
For each course in Course-Set
If Department-Code = Minor-Department-Code
Then
Add Credits to Minor-Credit-#.
Add corresponding Course data to MNCS.

.....

Process Number : 3.4.2
Process Name : Check-UM-Minor-Credits-Earned
Access File : None
Process Description :

 For each course in MNCS
 If UM-Course - "True"
 Then
 Add Credits to UM-Minor-Credit-#.

.....

Process Number : 3.5
Process Name : Check-Omnibus-Requirement
Access File : BA-Course-Set
Process Description :

 For each course in BA-Course-Set
 If Omnibus - "True"
 Then
 Add Credits to Omnibus-Credit-#.

.....

Process Number : 3.6

Process Name : Check-University-Writing-Exam

**Access File : Student-Examination-File,
Student-Profile.**

Process Description :

For each course in Course-Set
Add Credits to Credit-Sum.

Find GER-Year-Start, GER-Year-End in **Student-Profile** by
Student-ID.

If GER-Year-Start and GER-Year-End is - 1984 - 1985, or earlier
Then

Credit-Sum = Credit-Sum - 110.

Else if GER-Year-Start and GER-Year-End is - 1985 - 1986, or after
Then

Credit-Sum = Credit - Sum - 96.

If Credit - Sum < 0

Then

Less-UM-Credit-# = | Credit-Sum |

Else

Less-UM-Credit-# = 0.

If UM-Exam can be found in **Student-Examination-File** by
Student-ID.

Then

Set UM-Exam-Prompt to be "True"

Else

Prompt for new value if available.

.....

Process Number : 3.7
Process Name : Select-Student-Course
Access File : Student-Course-File
Process Description :

Find student Course-Set in **Student-Course-File** by Student-ID.

.....

Process Number : 4.1
Process Name : Update-Student-Profile
Access File : Student-Profile
Process Description :

Find Student-Profile-Set in **Student-Profile** by Student-ID.
For each Student-Profile-Data in Student-Profile-Set
Accept /Update field if desired.

.....

Process Number : 4.2
Process Name : Update-Date
Access File : Date-File
Process Description :

Find Raw-Student-Course-Set in **Raw-Student-Course-File** by Student-ID.
For each course in Raw-Student-Course-Set
Accept/Update course if desired.

.....

Process Number : 4.3
Process Name : Update-Student-Course
Access File : Raw-Student-Course-File
Process Description :

Find Raw-Student-Course-set in **Raw-Student-Course-File** by Student-ID.
For each course in Raw-Student-Course-Set
Accept/Update course if desired.

Accept new Raw-Student-Course.

.....

Process Number : 4.4

Process Name : Calculate-GPA

**Access File : Student-Course-File,
Student-Profile.**

Process Description :

Find Student-Course-Set in **Student-Course-File** by Student-ID.

For each course in Student-Course-Set

 Add Credits to Credits-Sum

 Case Grade

 A : Total-Sum = Total-Sum + Credits * 4

 B : Total-Sum = Total-Sum + Credits * 3

 C : Total-Sum = Total-Sum + Credits * 2

 D : Total-Sum = Total-Sum + Credits * 1

GPA = Total-Sum / Credits-Sum

Update GPA in **Student-Profile** by Student-ID.

.....

Process Number : 4.5

Process Name : Filter-Student-Course-File

**Access File : Raw-Student-Course-File,
Student-Course-File,
UM-Department-File.**

Process Description :

Find Raw-Student-Course-Set in **Raw-Student-Course-File** by Student-ID.

For each raw-course in Raw-Student-Course-Set

 Delete raw-course if

Grade - F.

 Repeat of class previously passed.

 Accounting for CSD credits in excess of 70, except

 CSD 219, 230, 251, 349, 350, 351, 360, 480, 481

 Any other 400 level course.

 "pass" course in excess of 60 credits.

For each raw-course which is not deleted in Raw-Student-Course-Set

 Convert Department into Department-Code by **UM-Department-File**.

 Add Department-Code and the rest course data into Student-Course-Set.

 Add Student-Course-Set into **Student-Course-File** by Student-ID.

.....

Process Number : 5.1

Process Name : Form-Student-Profile

Access File : Student-Profile

Process Description :

Find Student-Profile-Data in **Student-Profile** by Student-ID.

Dump each record in Student-Profile-Data into Student-Profile-Report-Form.

.....

Process Number : 5.2
Process Name : Form-Student-Course
Access File : Raw-Student-Course-File
Process Description :

Find Student-Course-Set in **Student-Course-File** by Student-ID.
Dump each course in Student-Course-Set into Raw-Student-Course-Report-Form.

.....

Process Number : 5.3
Process Name : Formulate-Student-Information-Report
Access File : Date-File
Process Description :

For each student report form
 Providing heading information into Formatted-Student
 Info-Report-Form.
Update new date in **Date-File** by the name of every **Student-
Master-File**.

.....

Process Number : 5.4
Process Name : Print-Formatted-Student-Information-Report
Access File :
Process Description :

Print Formatted-Student-Information-Report.

Appendix I

File Description

.....

File Name : Date-File

Alias :

File Contents :

1(Student-ID +
Current-Date)

Notes : The current date is changed when any one of the following files is updated.

e.g.

Student-File

Student-Course-File

Student-Examination-File

Student-Exemption-File

Student-Profile

.....

.....

File Name : Raw-Student-Course-File

Alias :

File Contents :

(Student-ID +
0(Raw-Student-Course))

.....

.....
File Name : Requirement-Master-File

Alias :

File Contents :

A set of the following files :

- Bachelor-Degree-Rule-File +
- Behavior-and-Social-Science-File +
- Capstone-File +
- Child-Develop-File +
- College-of-Arts-and-Science-File +
- CSD-File +
- CSD-400-File +
- Group1-Ethical-and-Human-File +
- Group2-Ethical-and-Human-File +
- Expressive-Arts-File +
- Foreign-Language-File +
- Historical-and-Cultural-Studies-File +
- Human-Anatomy-Physiology-File +
- Humanity-and-Arts-File +
- Literary-Artistic-Studies-File +
- Natural-Science-File +
- Natural-Science-Lab-File +
- Non-Western-Historical-and-Cultural-Studies-File +
- Oral-Communication-File +
- Required-Psyc-File +
- Science-and-Math-File +
- Statistics-File +
- Social-Science-File +
- Symbolic-System-Group-#-File +
- Western-Historical-and-Cultural-Studies-File +
- Written-Communication-File +
- W-Prefix-File

.....

.....
File Name : Student-Course-File
Alias :
File Contents :

(Student-ID +
o(Student-Course)

.....
.....
File Name : Student-Examination-File
Alias :
File Contents :

o(Student-ID +
o(Examination-Type))

Notes : Examination-Type, which has value 1 .. 30.
Each number represents for one examination name, which is a 30 character string.
1 for Math Placement Examination
2 for Writing Placement Examination, etc.

.....
.....
File Name : Student-Exemption-File
Alias :
File Contents :

o(Student-ID +
o(Exemption-Type))

Notes : Exemption-Type, which has value 1 .. 30.
Each number represents for one exemption name, which is a a 30 character string.
Example 1 for Physics-111-Exemption
2 for FLL-Exemption, etc.

.....

.....

File Name : Student-Master-File

Alias :

File Contents :

A set of the following files :

- Student-Course-File +
- Student-Examination-File +
- Student-Exemption-File +
- Student-Profile

.....

.....

File Name : Student-Profile

Alias :

File Contents :

- 0{ Student-Profile-Data +
- GPA }

.....

.....

File Name : Symbolic-System-File

Alias :

File Contents :

- 0{ Symbolic-Group +
- 1{ Course-ID }

Notes :

Symbolic-Group-# has value from 1 to 11 so far.

Symbolic-Group

Course-ID

1

CS 101, CS 102, CS 103

2

Ling 301, Ling 311, Ling 312

.....

.....

File Name : UM-Department-File

Alias :

File Contents :

1(Department-Code +
Department)

Notes :

e.g.	Department-Code	Department
	112	CS

.....

.....

File Format Name : Type-One

File Contents :

0(Course-ID +
Quarter-Offered +
Year-Offered)

Files are :

- Behavior-and-Social-Science-File,
- Group1-Ethical-and-Human-File,
- Group2-Ethical-and-Human-File,
- Historical-and-Cultural-Studies-File,
- Humanity-and-Arts-File,
- Literary-Artistic-Studies-File,
- Natural-Science-File,
- Natural-Science-Lab-File,
- Non-Western-Historical-and-Cultural-Studies-File,
- Science-and-Math-File,
- W-Prefix-File,
- Western-Historical-and-Cultural-Studies-File

.....



File Format Name : Type-Two

File Contents :

o(Course-ID)

Files are :

- Child-Develop-File,
- Expressive-Arts-File,
- Foreign-Language-File,
- Human-Anatomy-Physiology-File,
- Oral-Communication-Ability-File,
- Statistic-File,
- Social-Science,
- Writing-Communication-Ability-File



File Format Name : Type-Three

File Contents :

17(Restrict-Credit-Number +
{ Type-One })

File is :

Bachelor-Degree-Rule-File



File Format Name : Type-Four

File Contents :

o(Department-Code)

Files are :

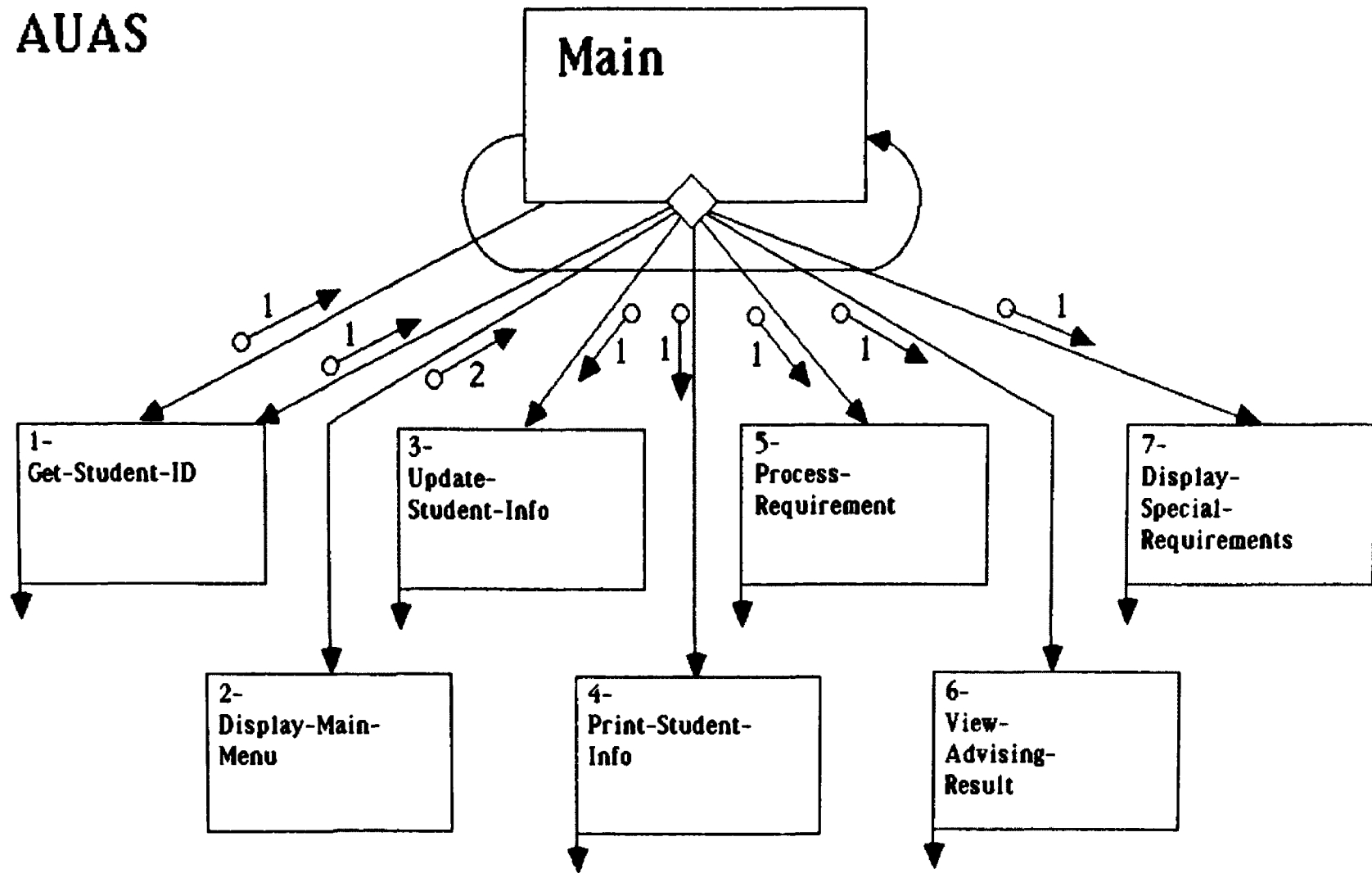
College-of-Arts-Science-File



Appendix J

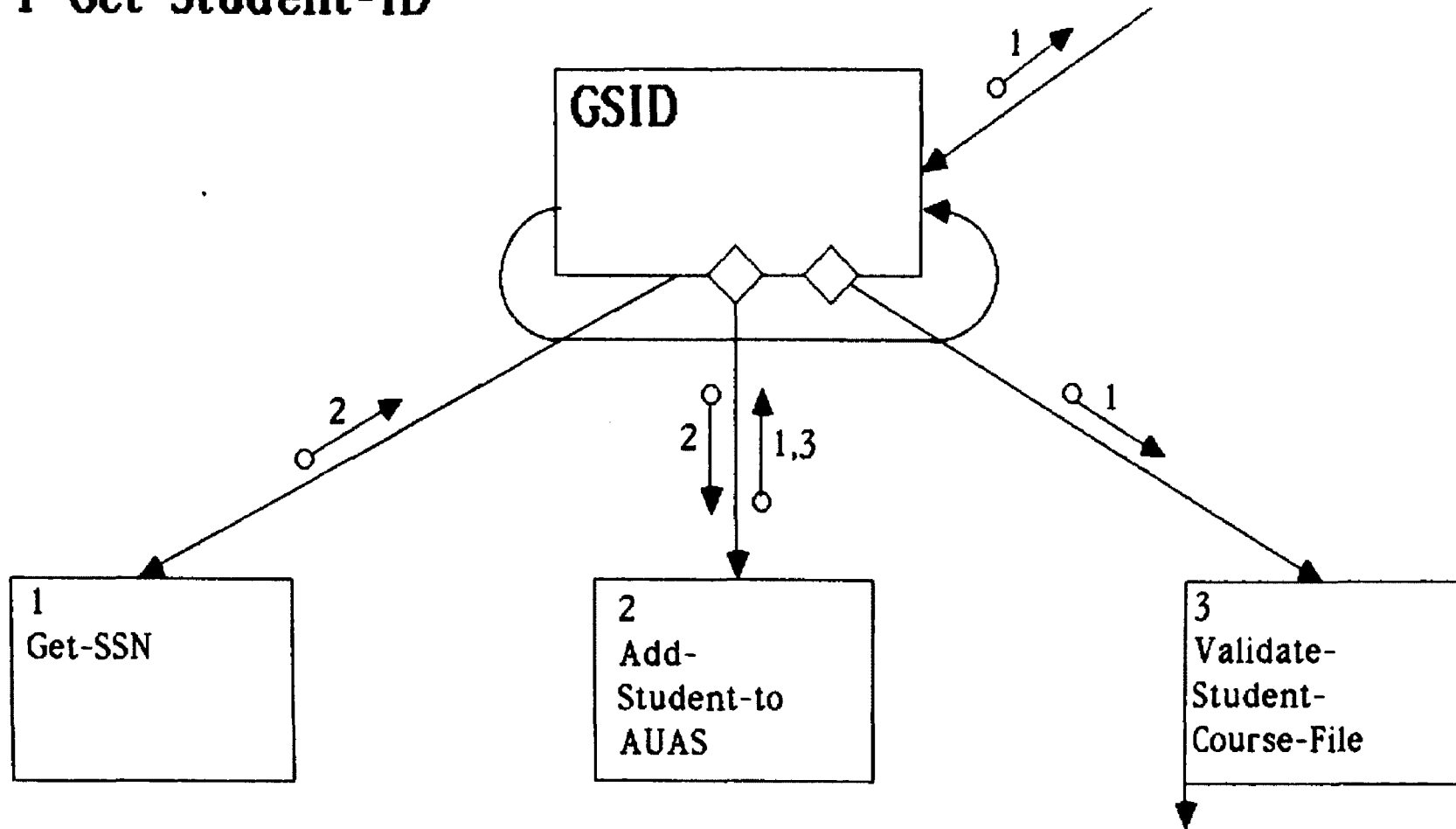
Structure Chart

AUAS



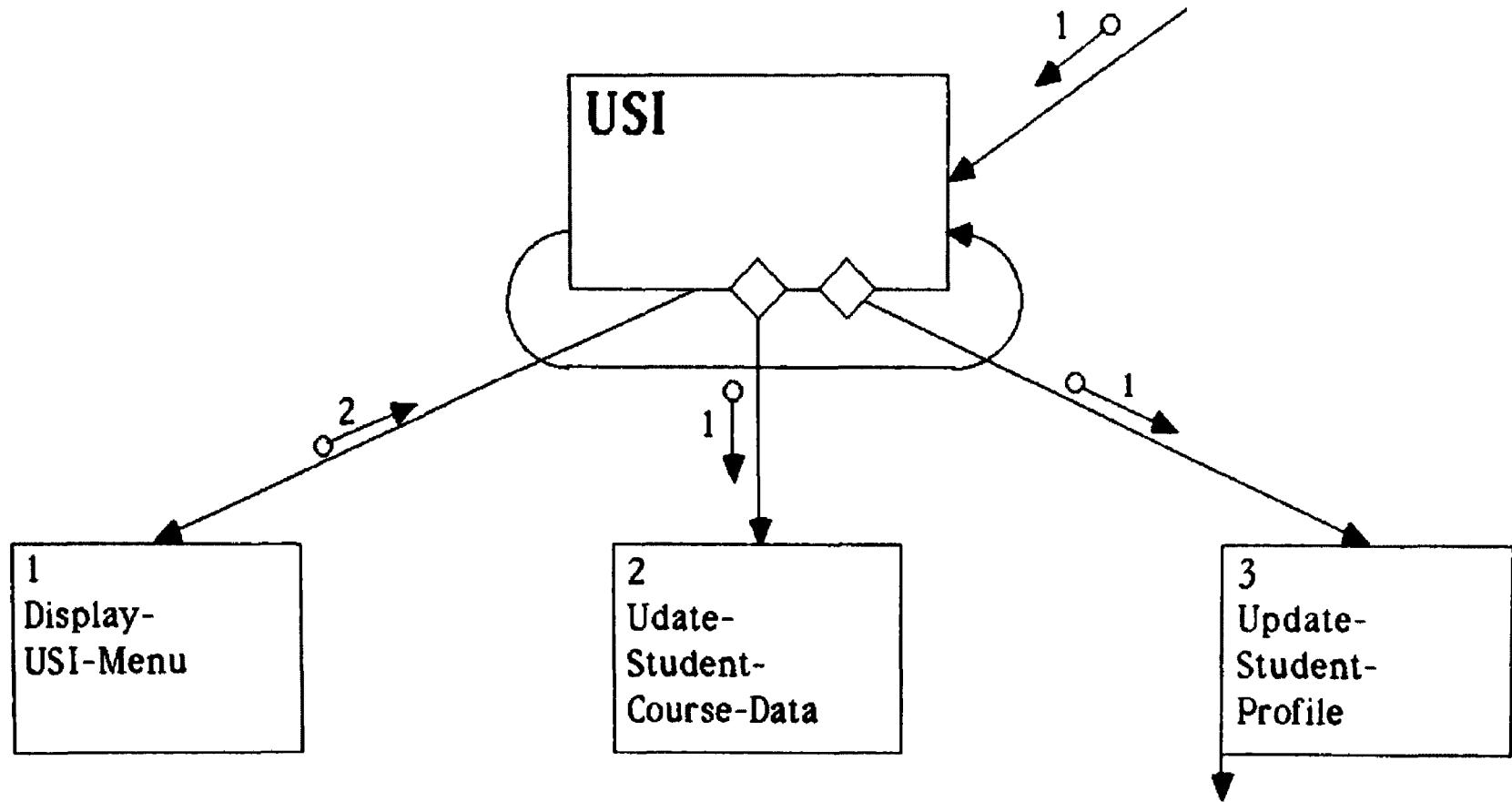
- 1 = Student-ID
- 2 = Menu-Choice

1-Get-Student-ID



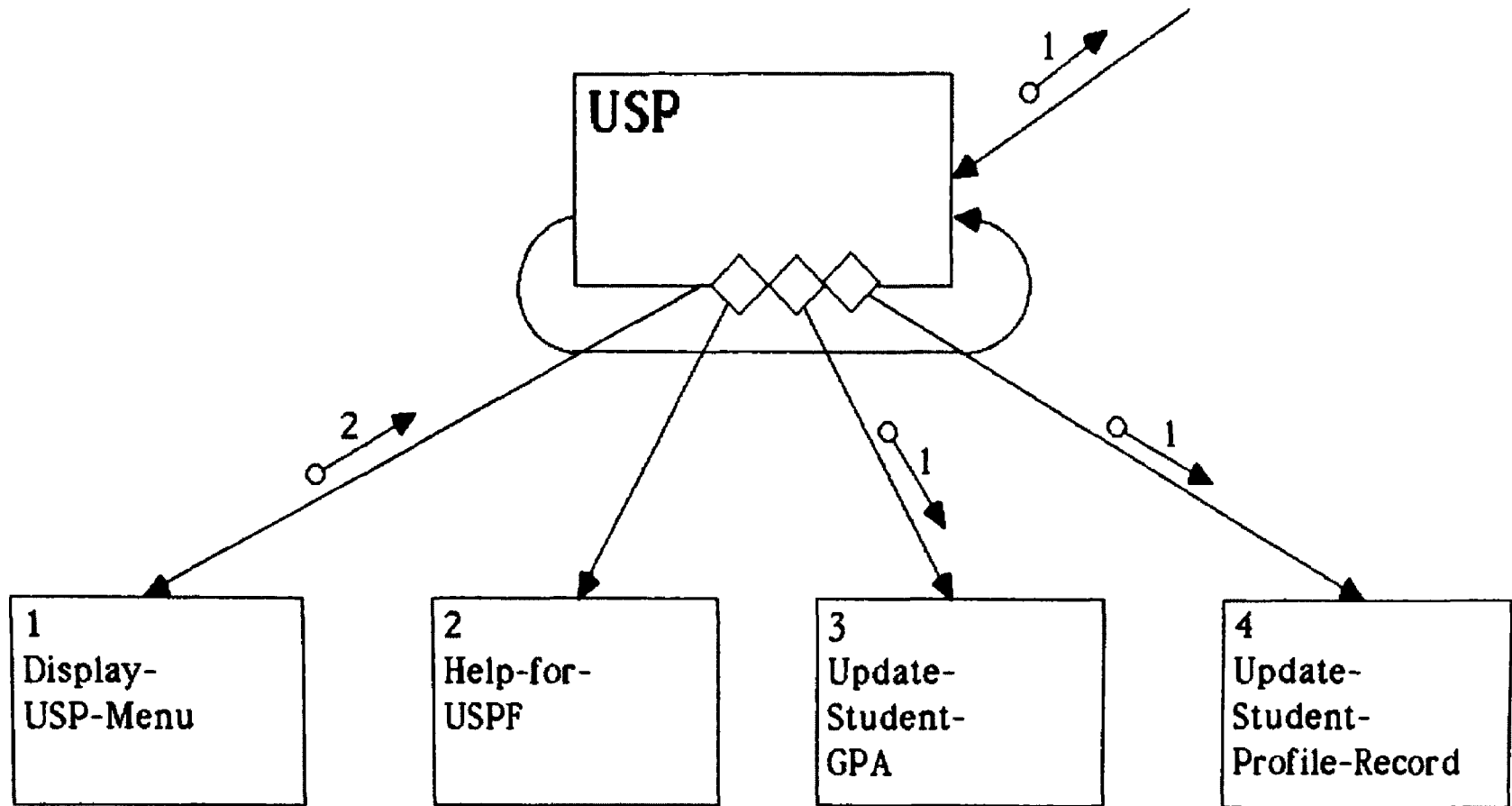
- 1 - Student-ID
- 2 - SSN
- 3 - Empty-Student-Data-Flag

3-Update-Student-Info



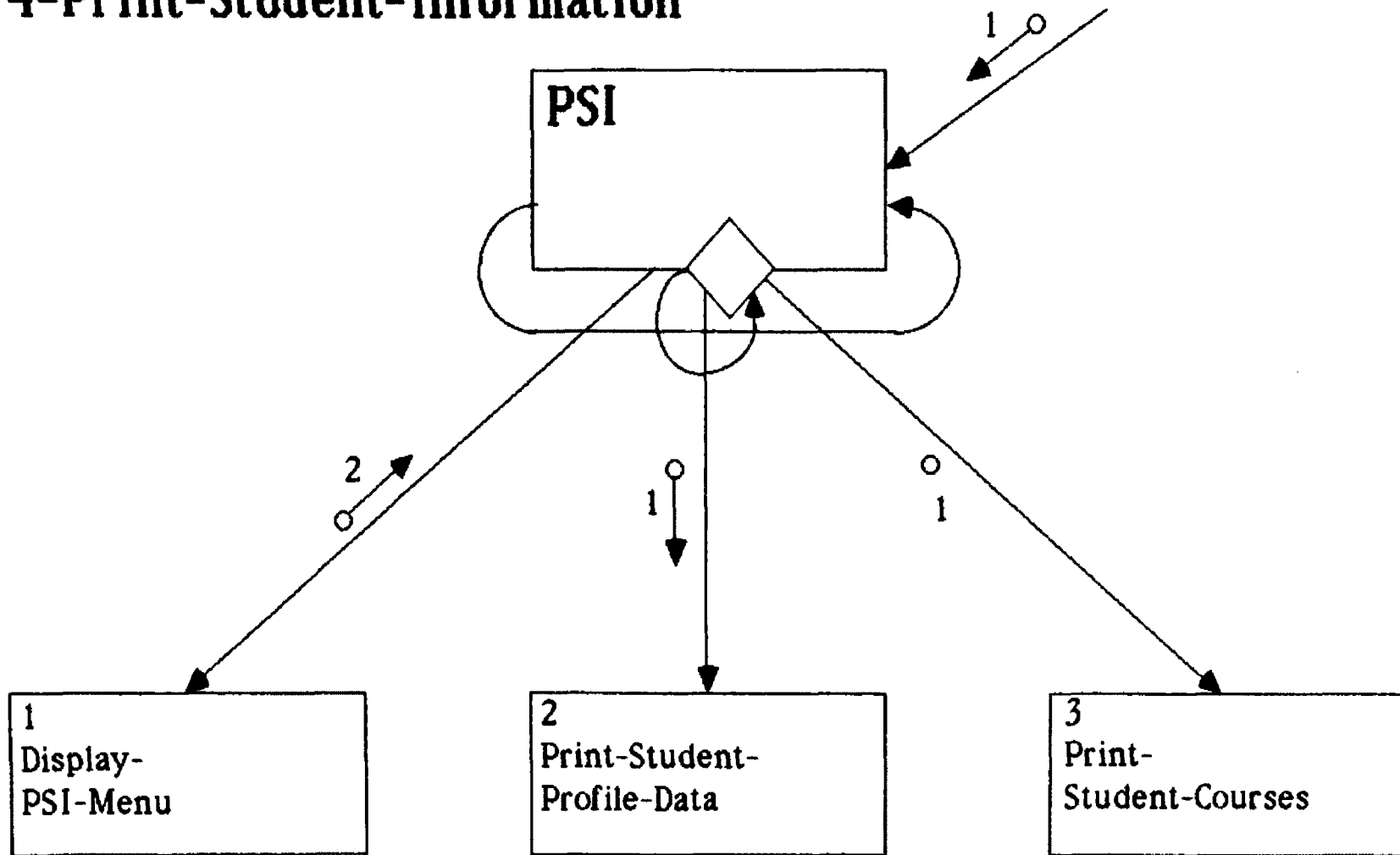
1 = Student-ID
2 = USI-Menu-Choice

33-Update-Student-Profile



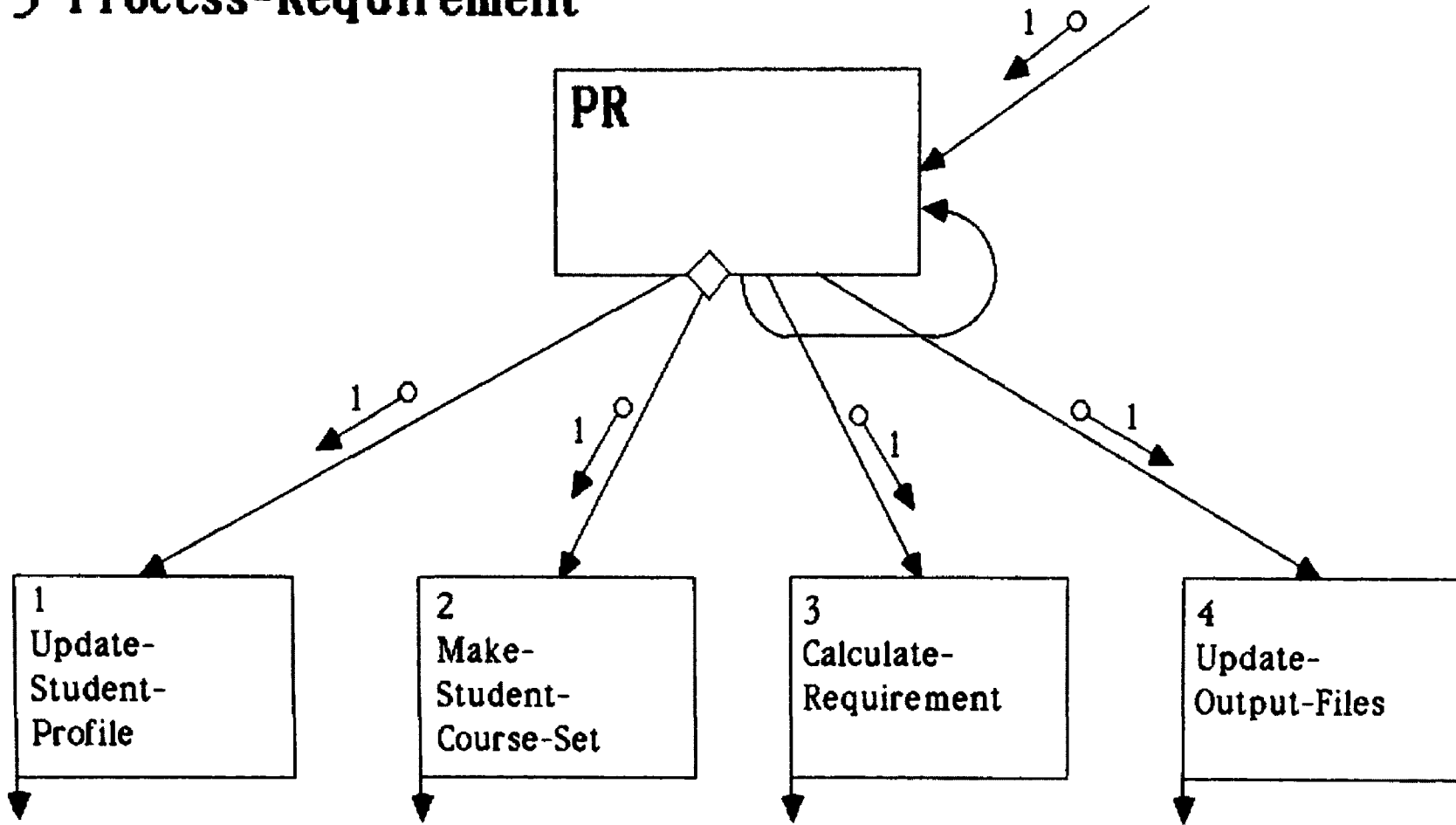
1 = Student-ID
2 = Menu-Choice

4-Print-Student-Information



1 = Student-ID
2 = USI-Menu-Choice

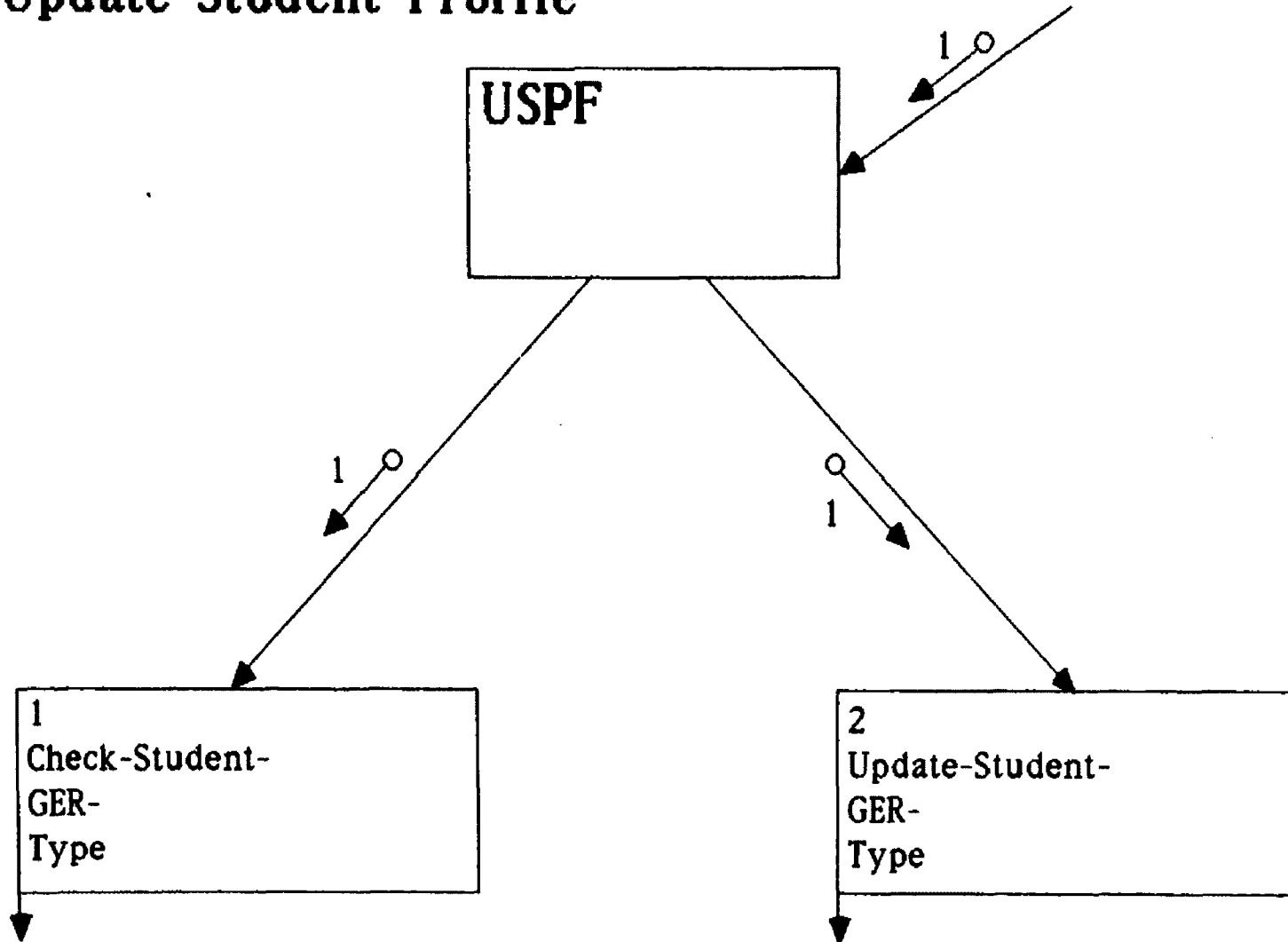
5-Process-Requirement



1 = Student-ID

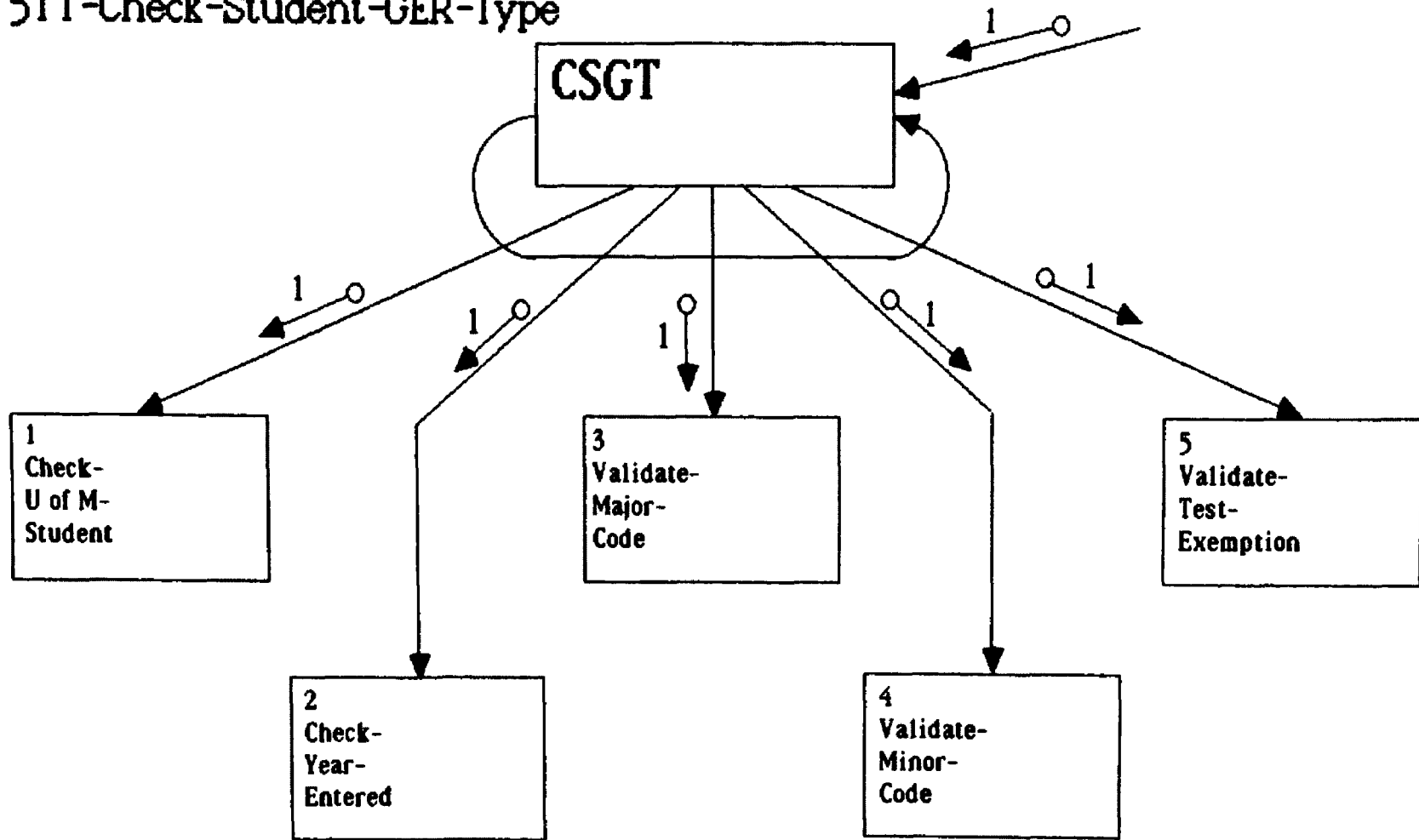
2 = USI-Menu-Choice

51-Update-Student-Profile



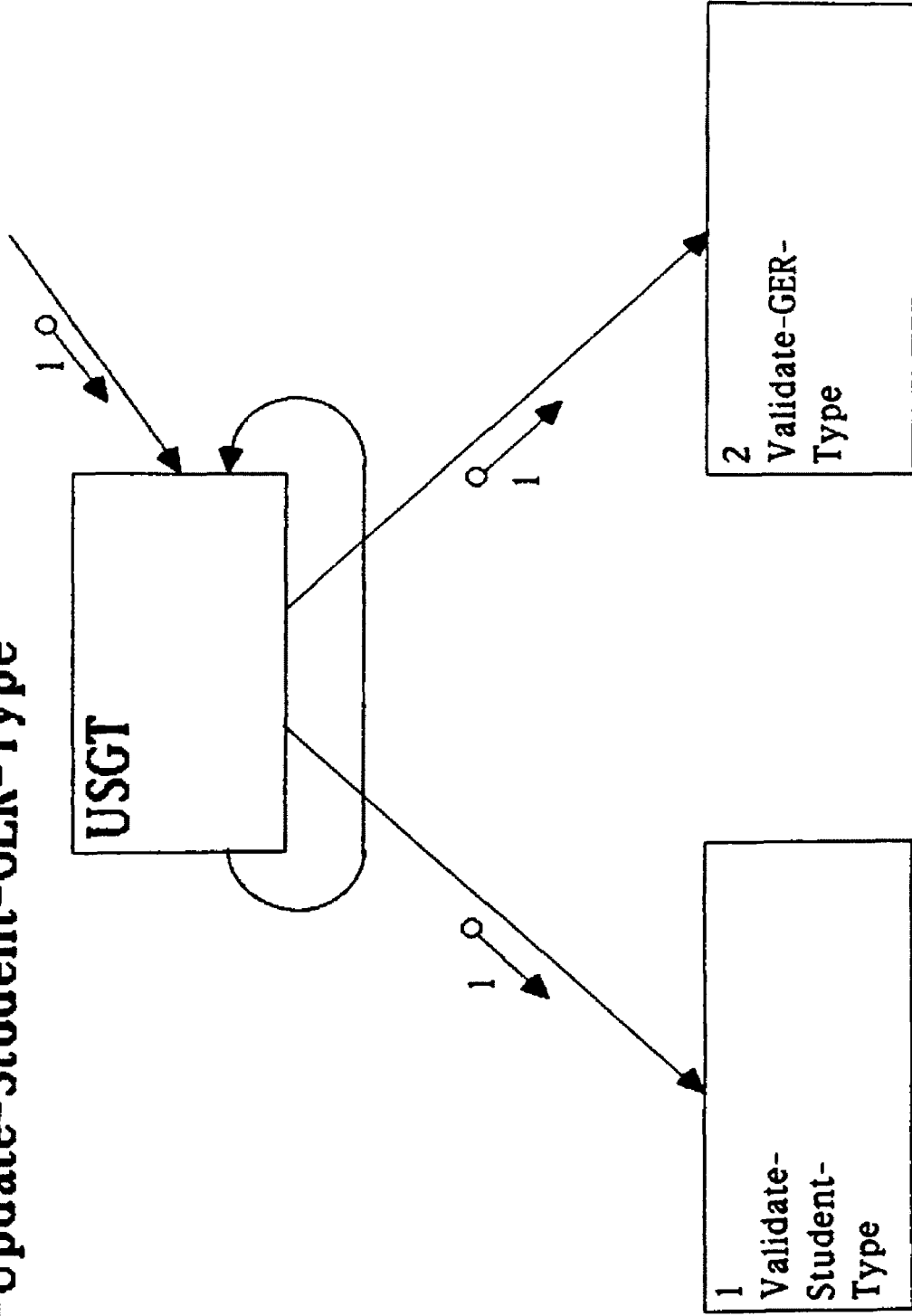
1 - Student-ID

511-Check-Student-GER-Type

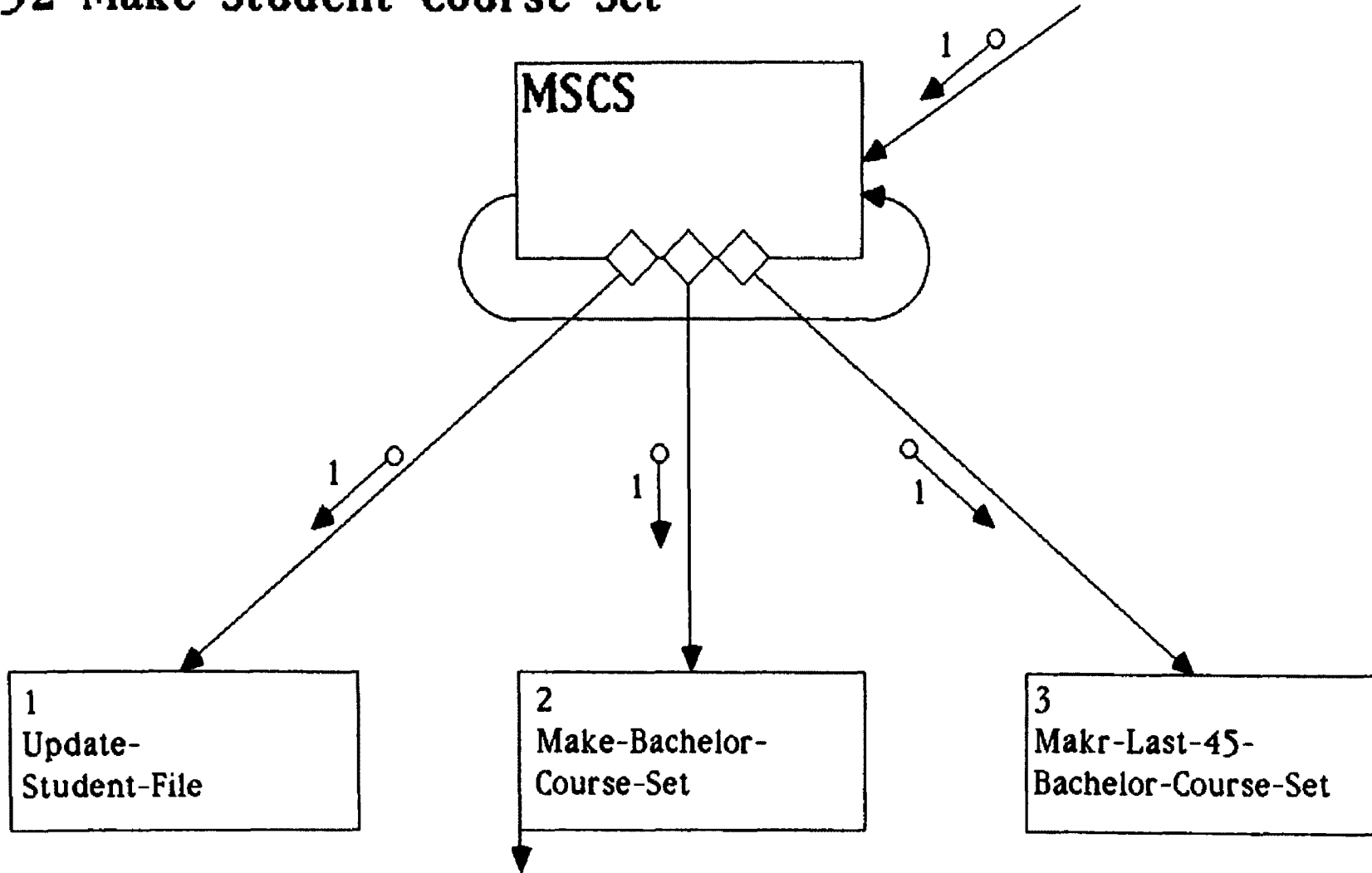


1 - Student-ID

512-Update-Student-GER-Type

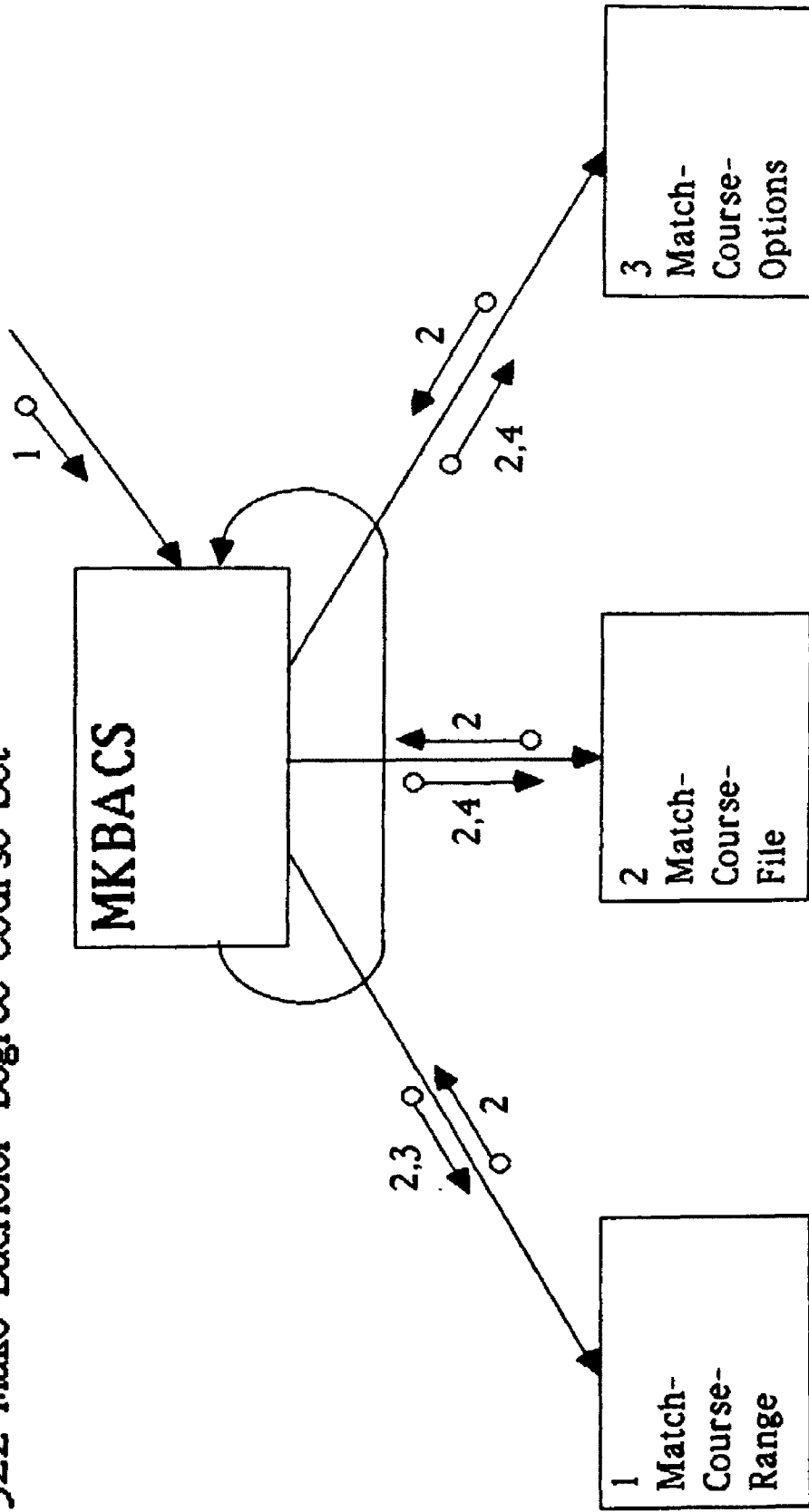


52-Make-Student-Course-Set



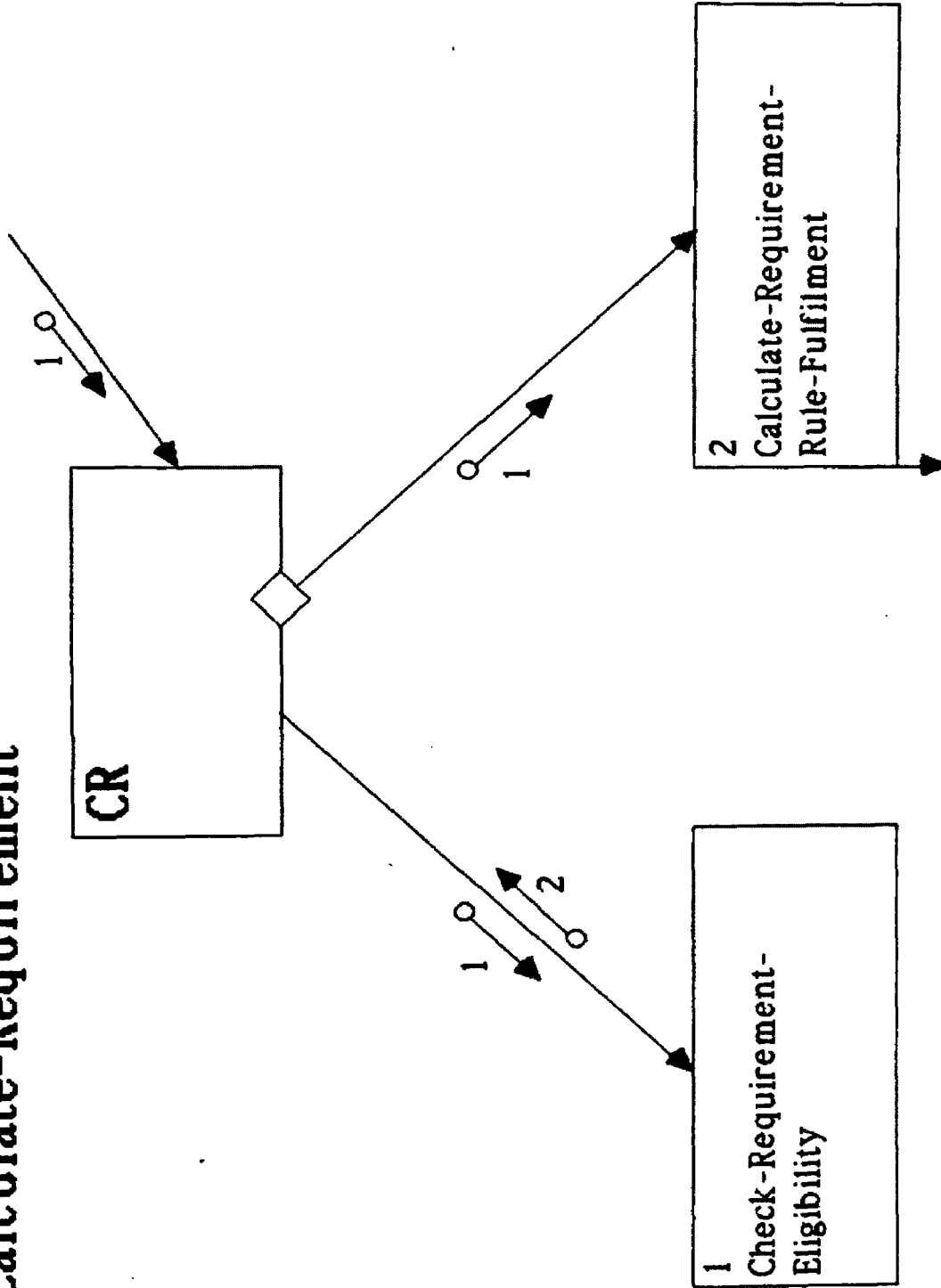
1 = Student-ID
2 = Menu-Choice

522-Make-Bachelor-Degree-Course-Set



- 1 = Student-ID
- 2 = Selected-Course-Set
- 3 = Course-Range
- 4 = Course-Files-Name
- 5 = Course-Options

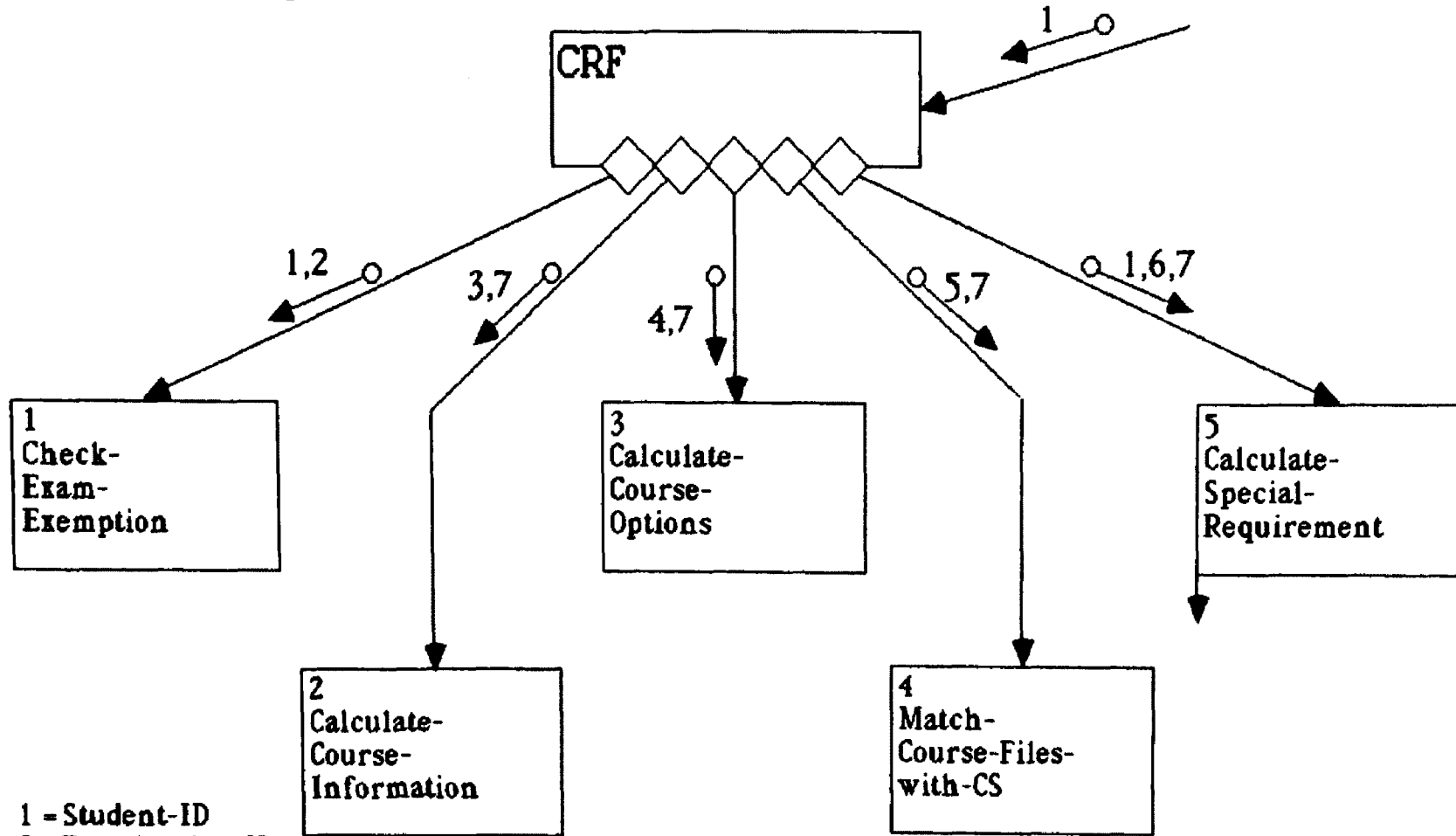
53-Calculate-Requirement



1 - Student-ID

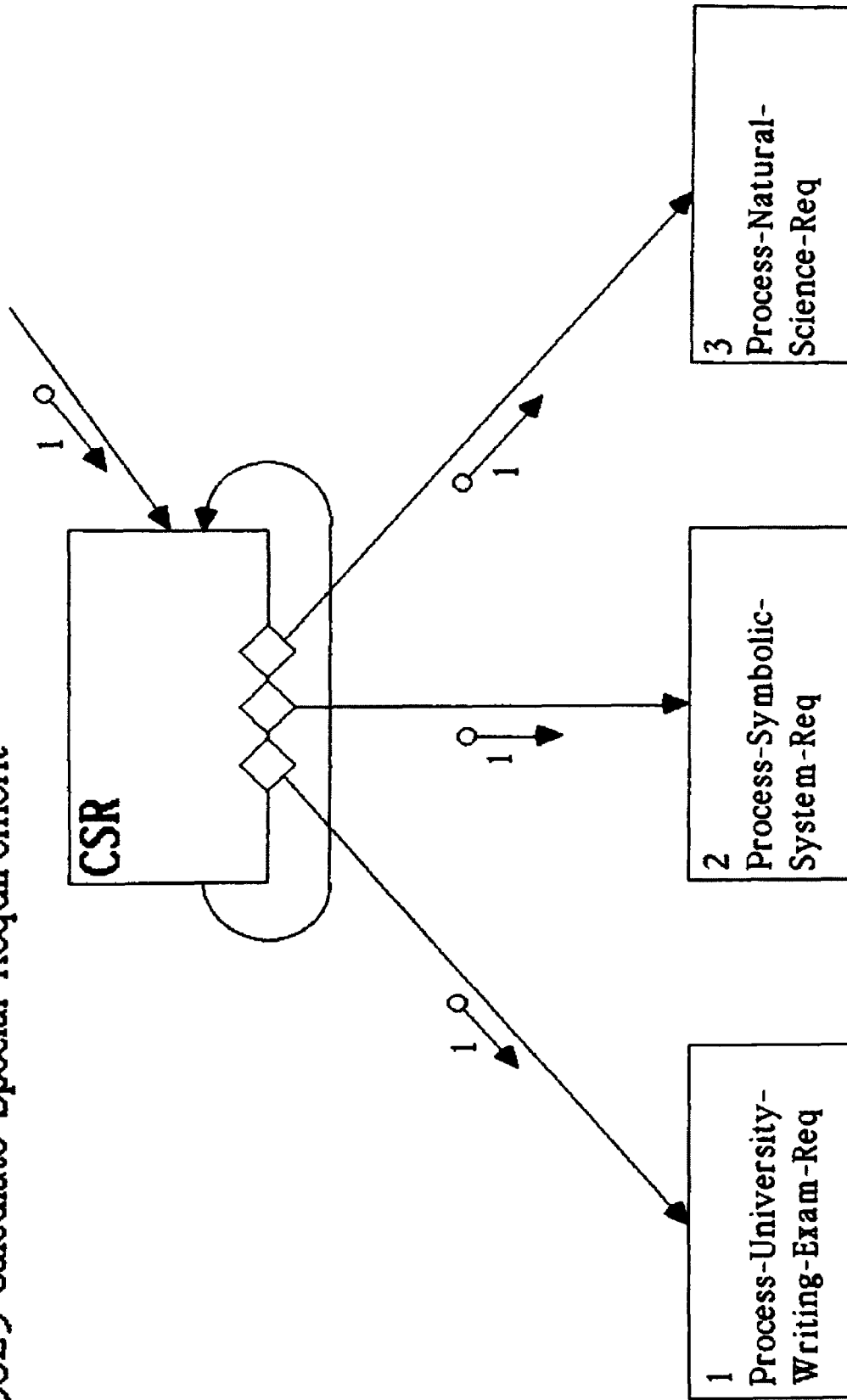
2 - Flag

532-Calculate-Requirement-Fulfillment



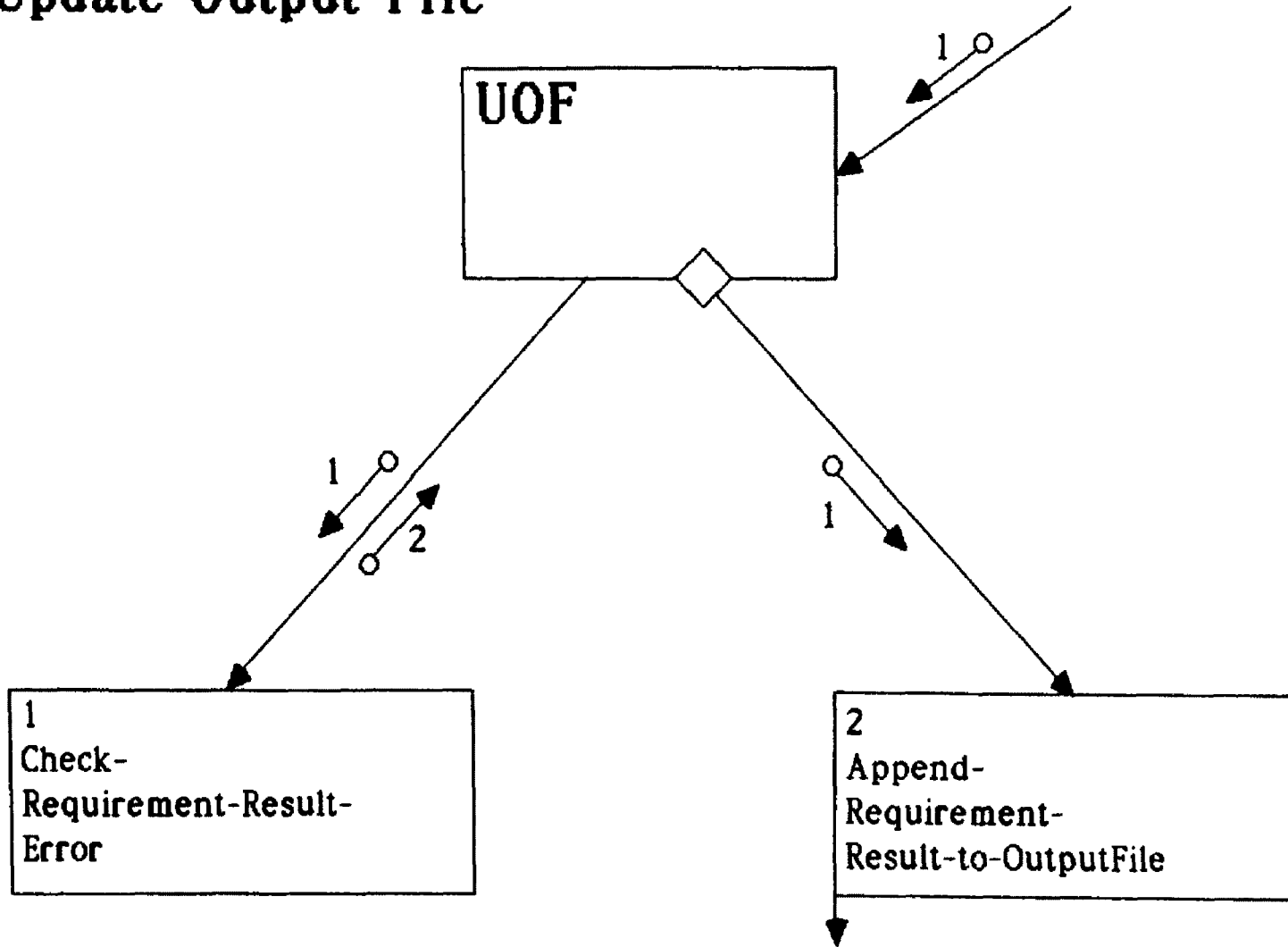
- 1 = Student-ID
- 2 = Examination-Name
- 3 = Course-Information
- 4 = Course-Options
- 5 = Match-Course-File-Names
- 6 = Special-Requirement
- 7 = Selection-File

5325-Calculate-Special-Requirement



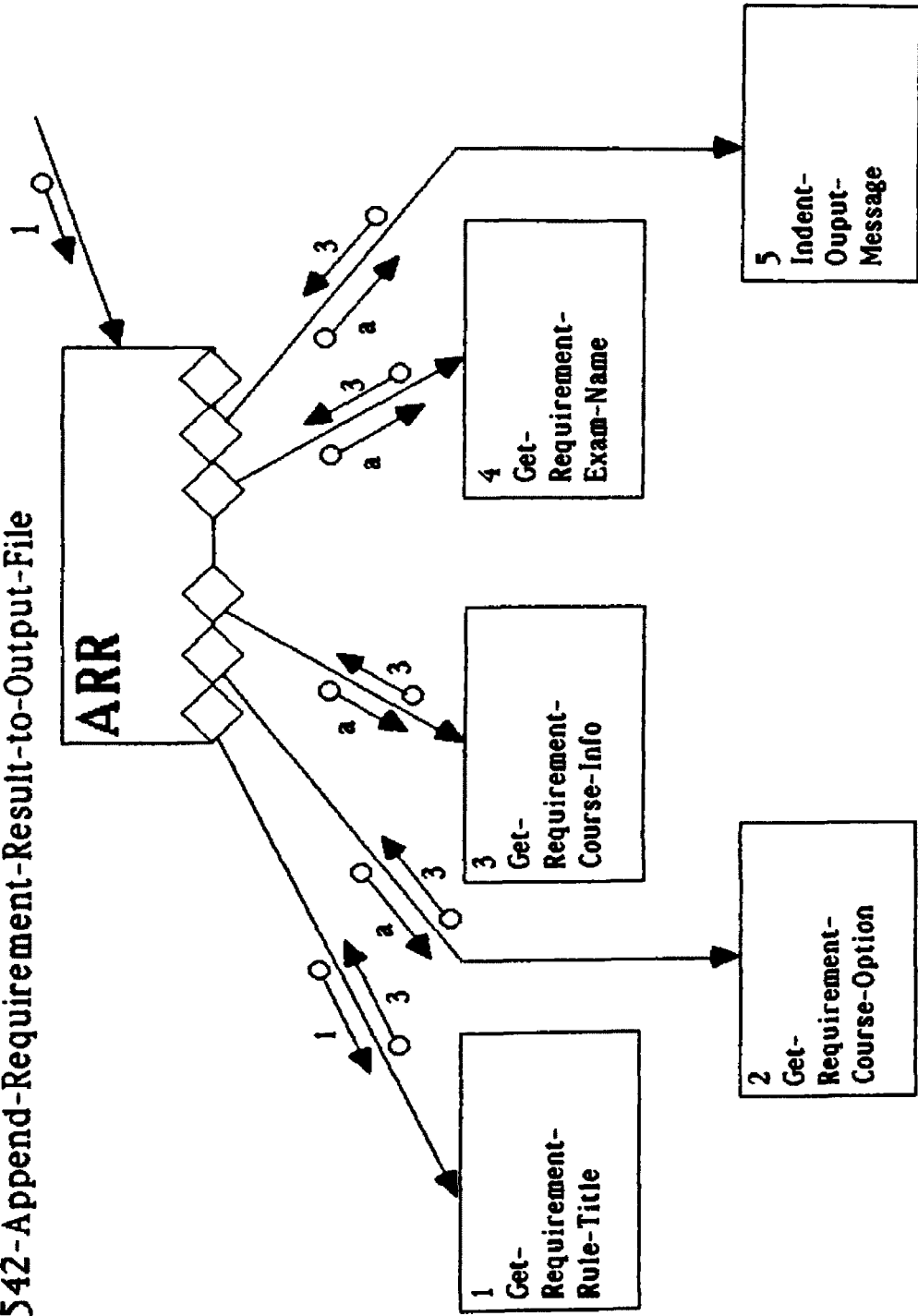
1 - Student-ID

54-Update-Output-File



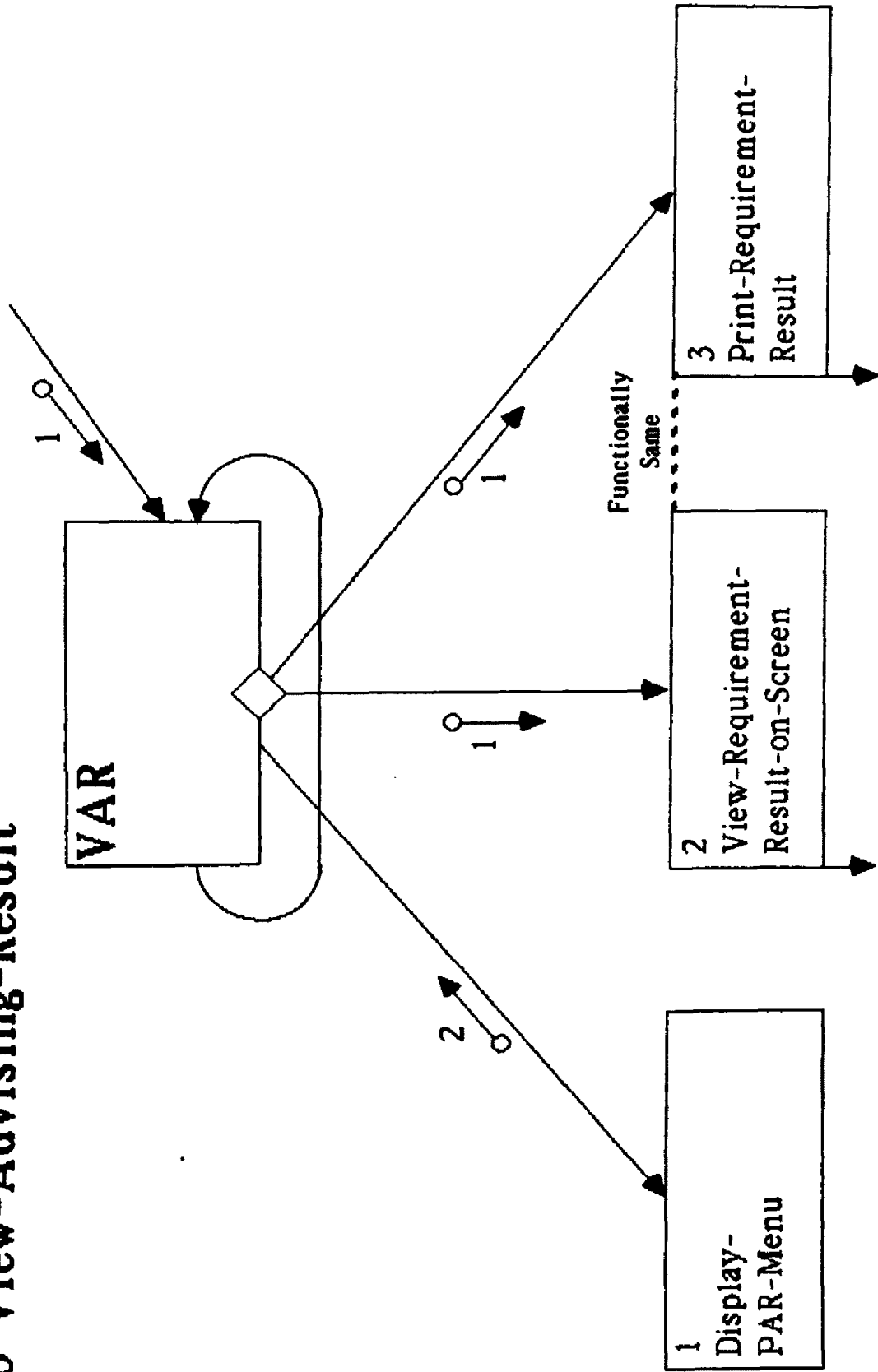
1 = Student-ID
2 = Flag

542-Append-Requirement-Result-to-Output-File



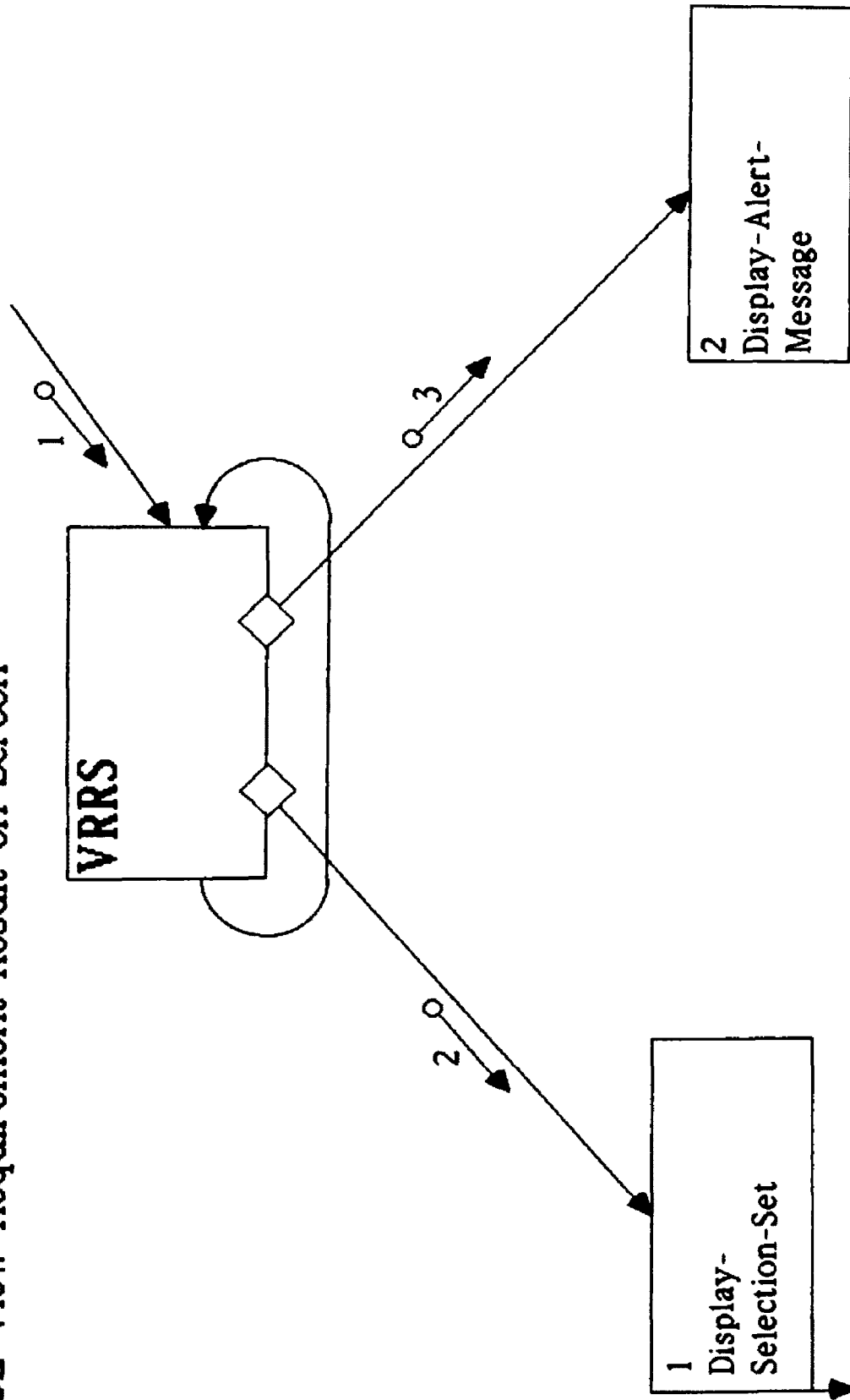
- 1 - Student-ID
- 2 - Output-File-Record
- 3 - Requirement-Output-String(Screen/Print)
- a - 2, 3

6-View-Advising-Result



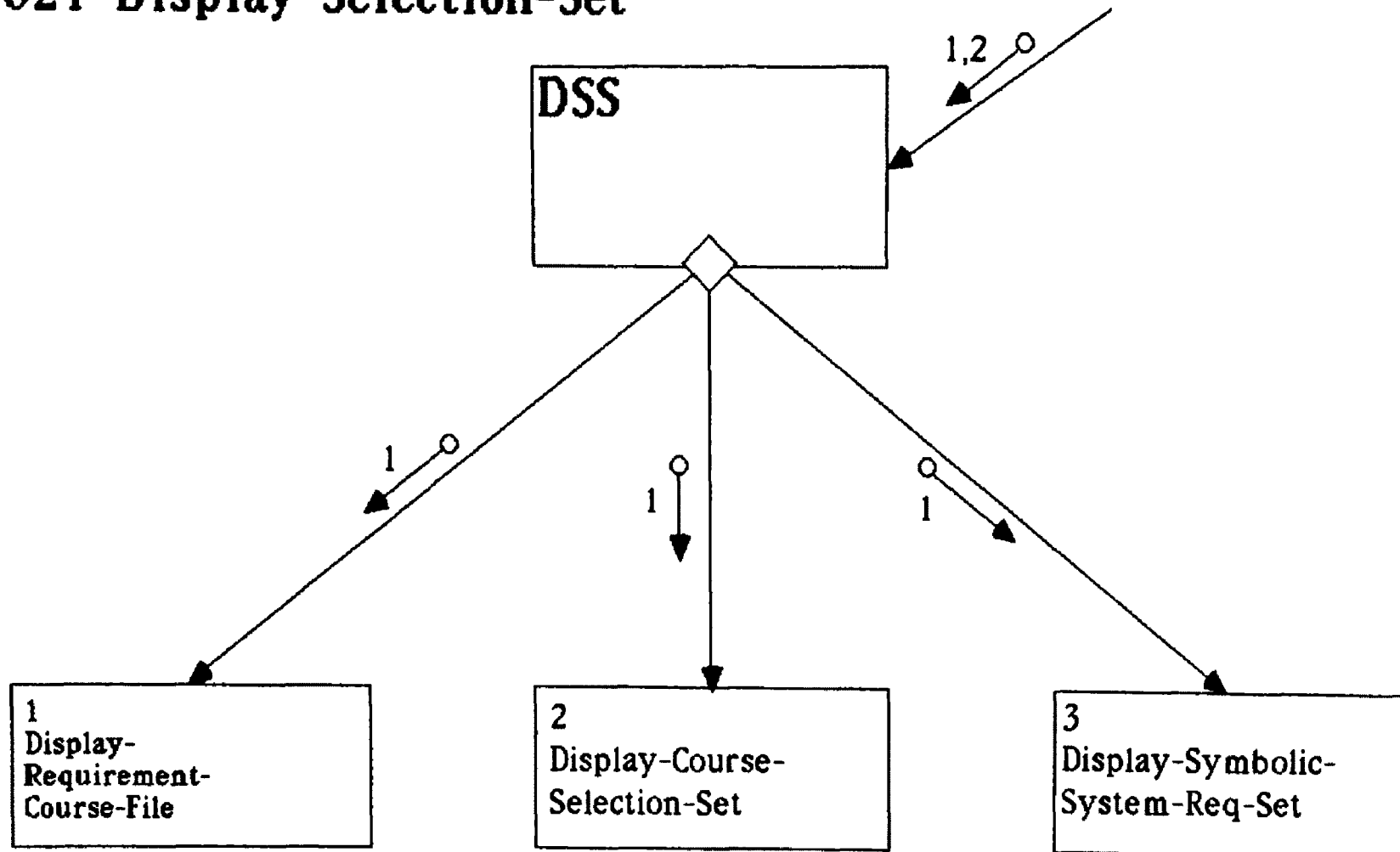
- 1 = Student-ID
- 2 = Menu-Choice

62-View-Requirement-Result-on-Screen



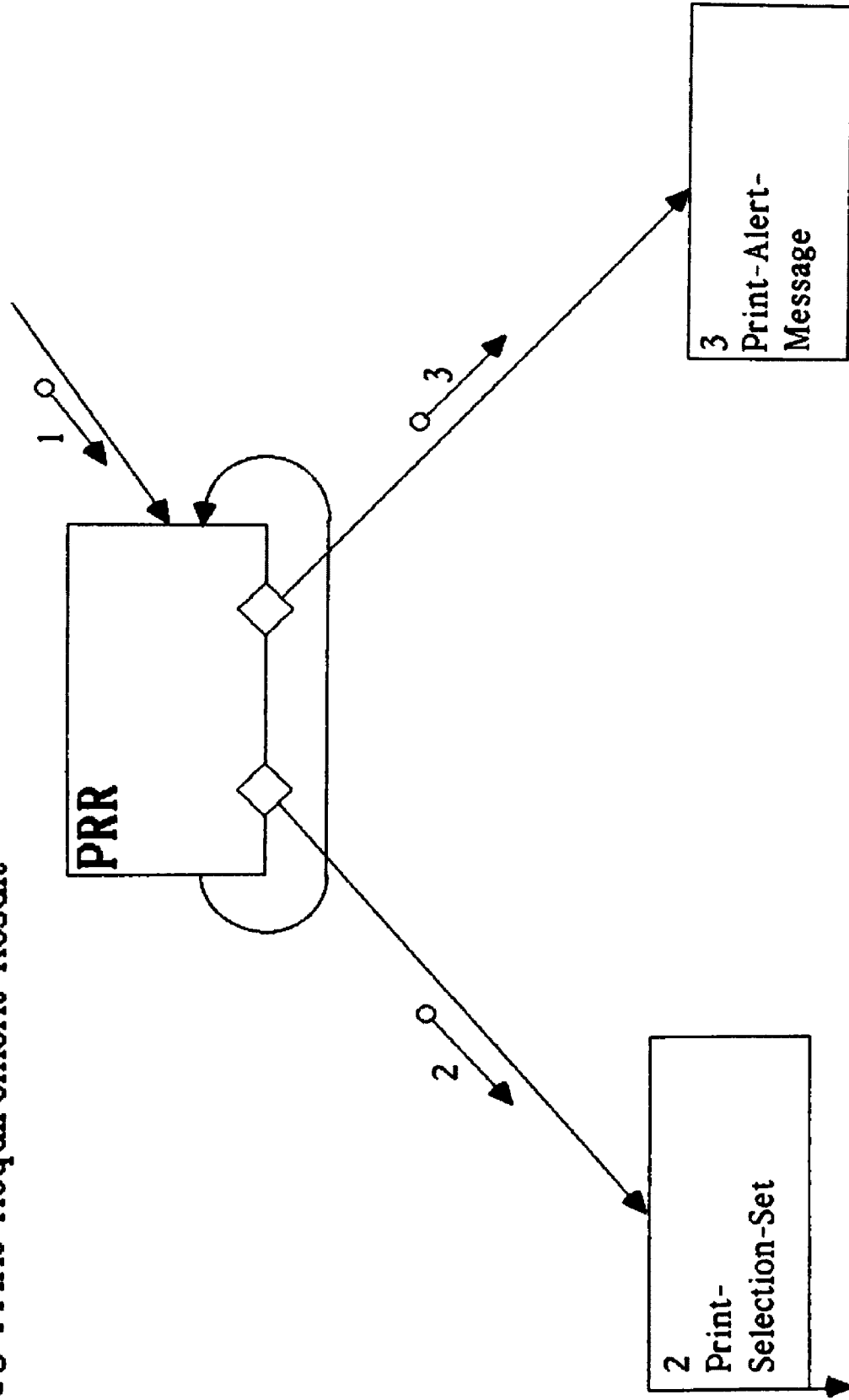
- 1 - Student-ID
- 2 - Selection-Set-File-Name
- 3 - Unselected-BACS

621-Display-Selection-Set



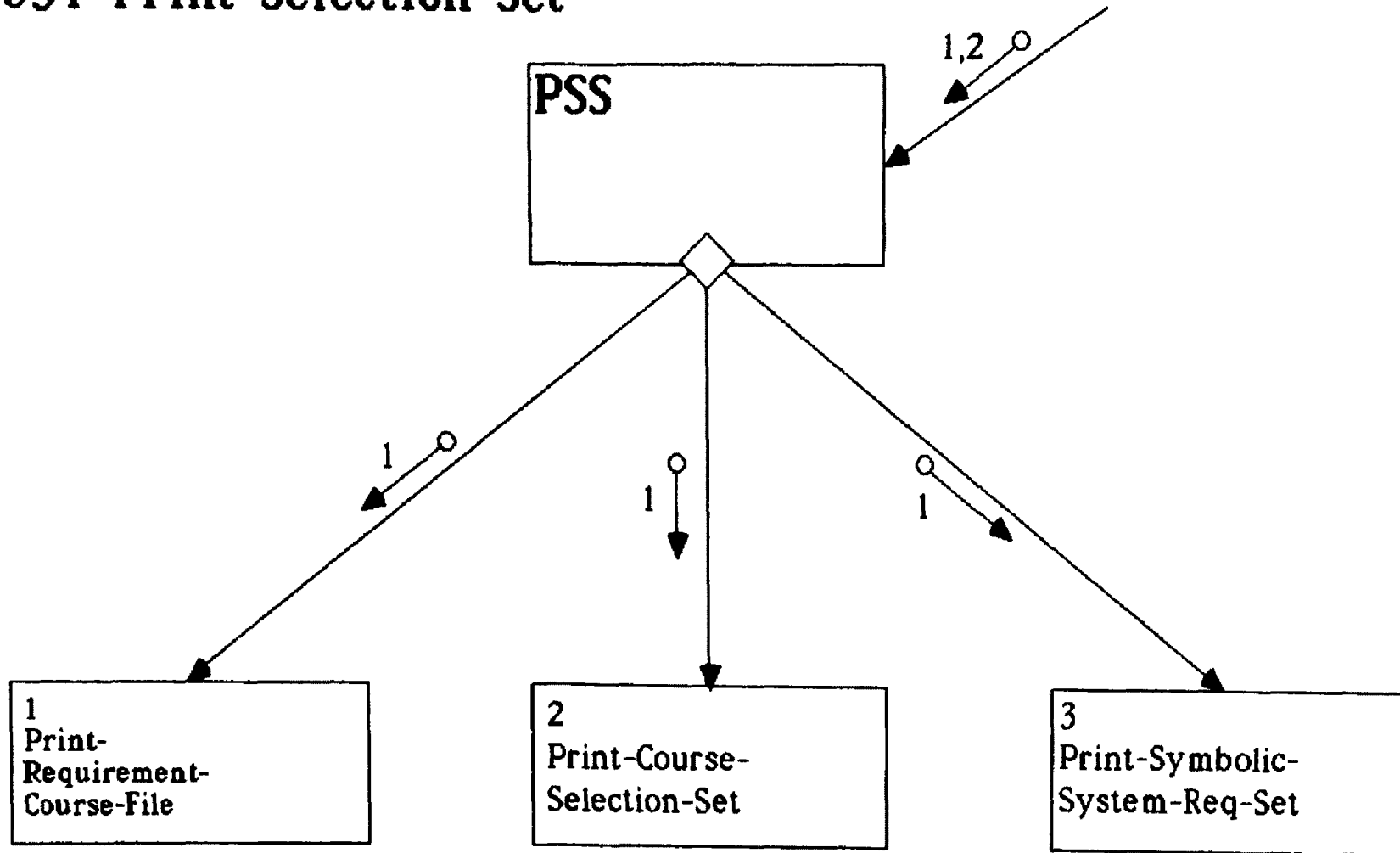
1 - Selection-Set-Name
2 - Selection-Set-Type

63-Print-Requirement-Result



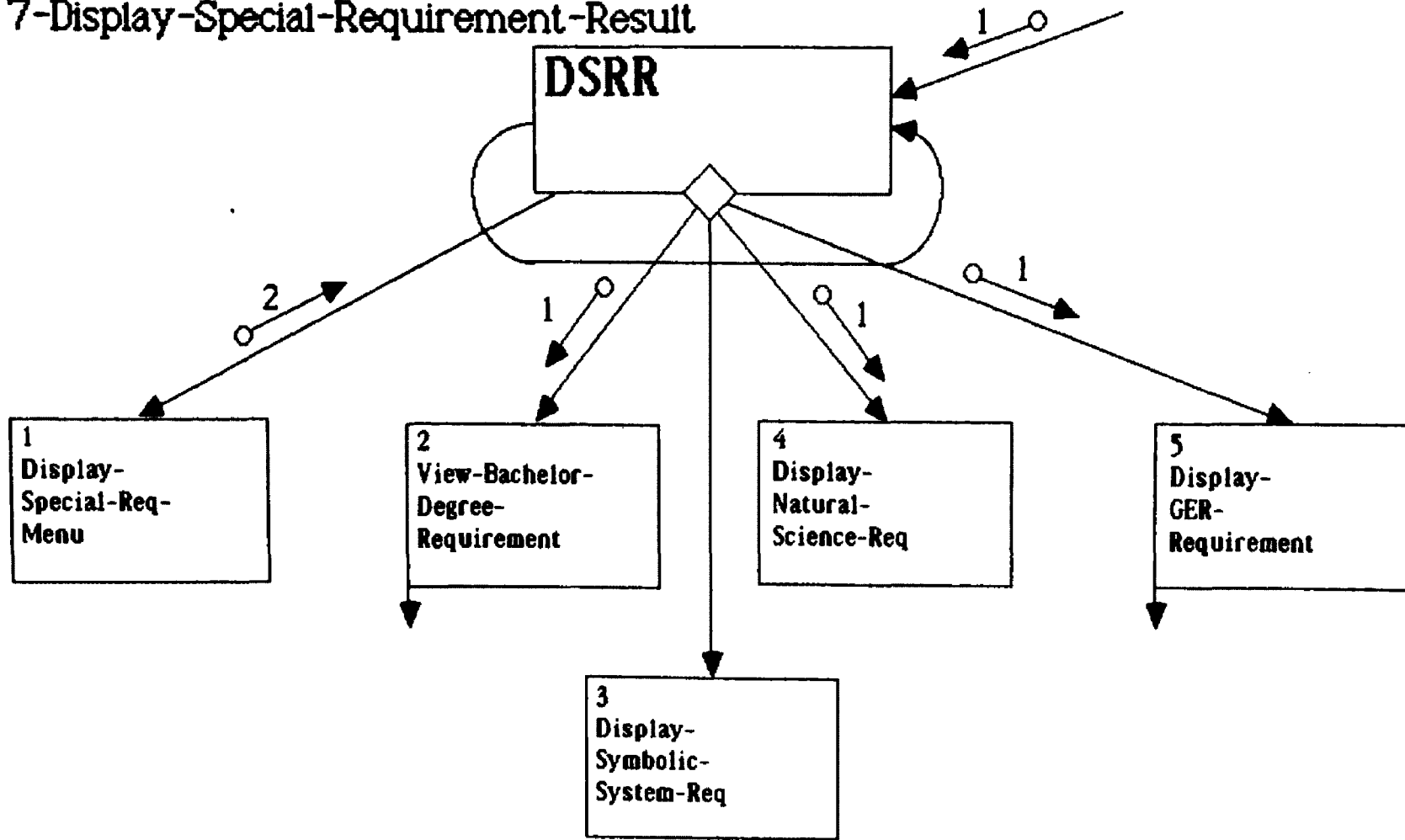
- 1 = Student-ID
- 2 = Selection-Set-File-Name
- 3 = Unselected-BACS

631-Print-Selection-Set



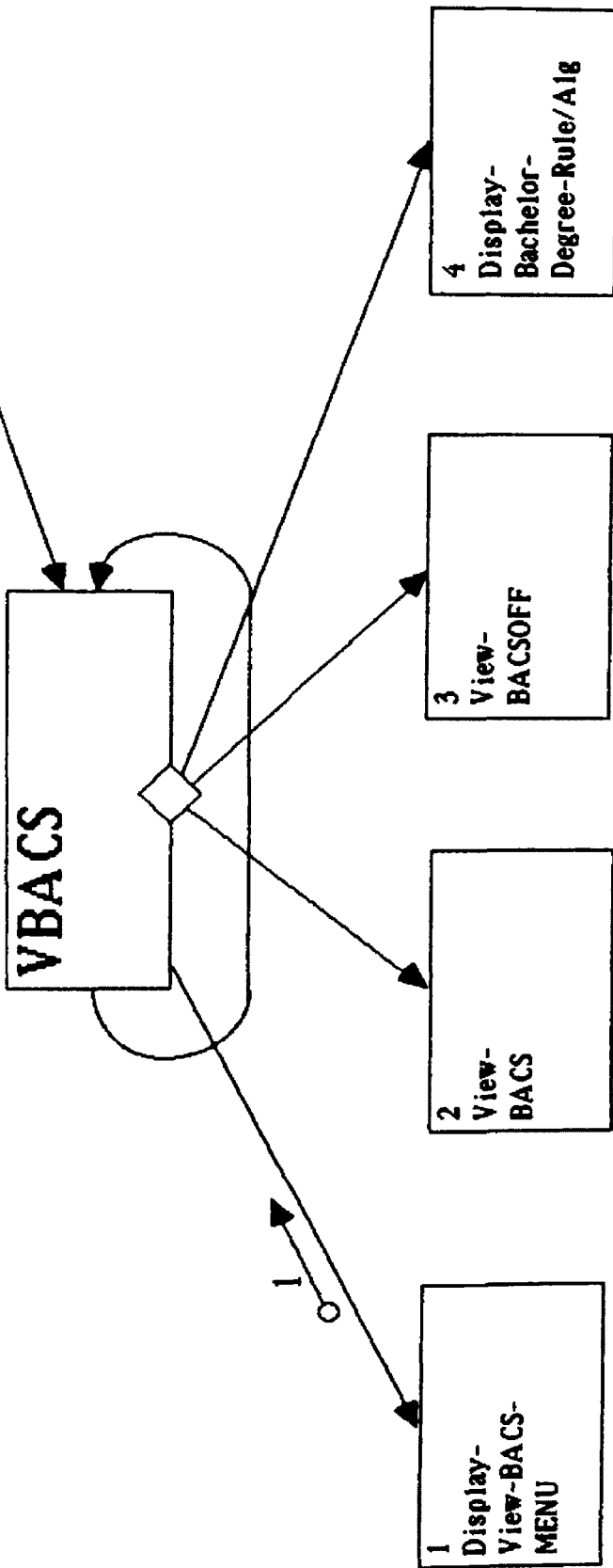
1 = Selection-Set-Name
2 = Selection-Set-Type

7-Display-Special-Requirement-Result



1 = Student-ID
2 = Menu-Choice

72-VIEW-Bachelor-Degree-Course-Set



75-Display-GER-Requirement

