University of Ottawa : CSI4124/SYS5110
Foundations of Modelling and Simulation

Case Study 3: FSB Coin System

Date

Ali Mowazi
Rahman Suleiman
Peter Wu
Su Zhang
Ryan Warnock

1. Problem Description

Problem Statement

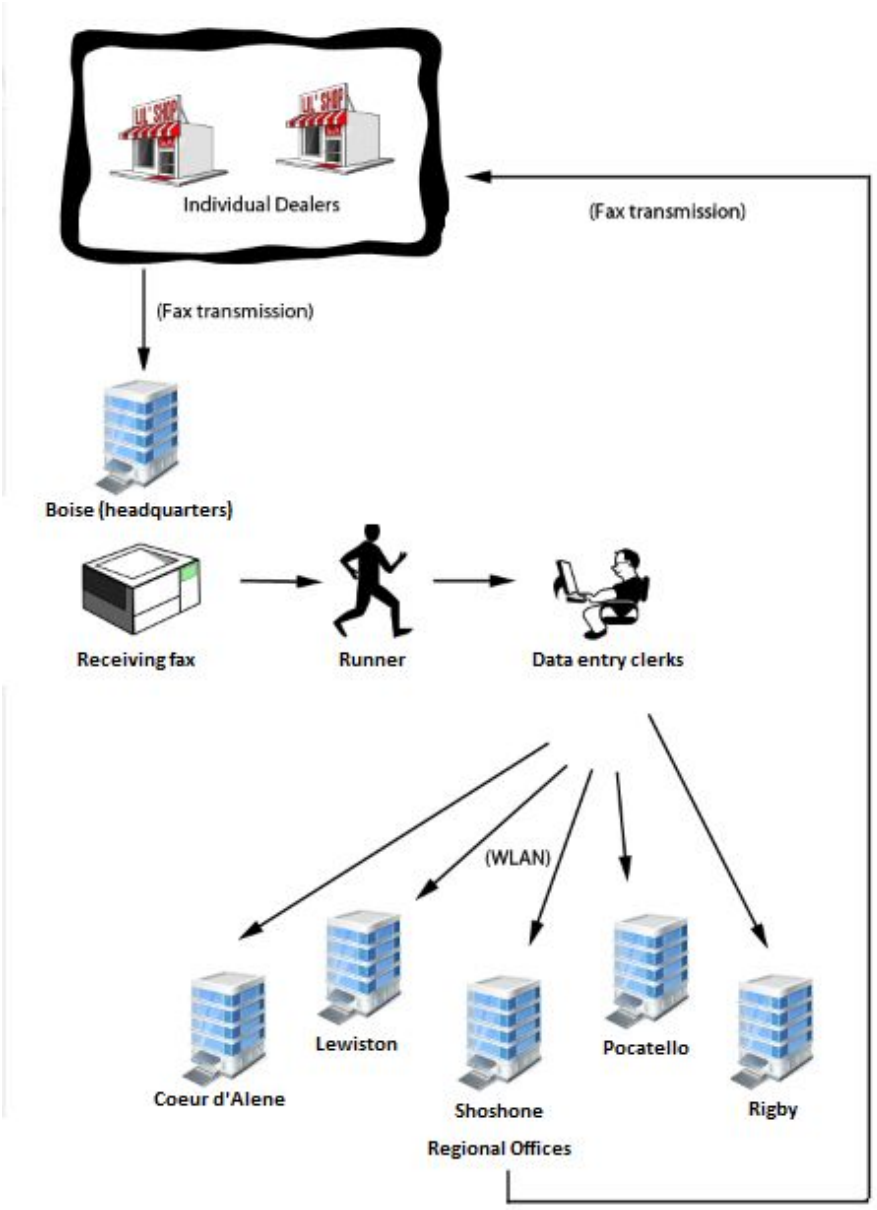
A consumer lending software package is being implemented by the First Security Bank (FSB). Consumer loans depend on on this system and the majority of these loans are auto loans. The implemented system has been operating for approximately six months and fails to meet the turnaround time of thirty minutes for auto loans. The turnaround time is measured by the time it takes from the dealer sending the application to receiving a decision.

There is currently an error rate of 10% which exceeds the expected error rate of 5% in the data entry process. Additionally, staffing seems to be an issue in some regions and the processing of data is taking longer than expected. Todd from First Security Bank believes that if the high error rate and processing time issues can be corrected and the staff increased, the goal time of thirty minutes or less can be met. As the industry is very competitive, it is vital that this target is reached.

System Under Investigation (SUI) Details

FSB Coin Organization:

First Security Bank of Idaho is the second largest bank in the state of Idaho with branches throughout the state. The bank is a full-service bank providing a broad range of banking services. Consumer loans and, in particular, auto loans make up an important part of these services. The bank is part of a larger system covering most of the intermountain states and its headquarters are in the city of Boise (the capital of Idaho).



The organisation consists of:

1. Automobile dealership: Customers complete an application with the purpose of borrowing money to purchase a car

2. Headquarters: Located in Boise, the headquarters have eight fax machines receiving fax transmission from the auto dealers. A runner distributes the application to one of eight data entry clerks.

3. Regional Offices: Six offices located in different cities, one of which is the headquarters at Boise, with their own staff of loan officers.

4. Personnel:

   - Data entry clerks work at the headquarters and enter data of the application, then send the application to one of the regional offices, as determined by the geographic location of the automobile dealer the application came from.

   - Loan officers work at each regional office. They process the loan, make a decision, and send the response back to the automobile dealer. Loan officers are also responsible for correcting any errors made by data entry clerks.

   The distribution of loan officers in each district is as follows:

District Number	Headquarters	Number of officers
1	Coeur D'Alene	4
2	Lewiston	2
3	Boise	6
4	Shoshone	3
5	Pocatello	2
6	Rigby	2

Processing Loans:

Phase 1: The process begins when a loan request has been completed at an individual dealership and is transmitted to the bank headquarters via fax.

Phase 2: FSB headquarters has eight (8) fax machines capable of receiving eight (8) simultaneous faxes. It is expected that this is sufficient capacity such that there is no queue time for receiving faxes. Once the fax is received it is picked up by a runner who distributes the fax to one of eight available data entry clerks. The runner completes this task quickly but a queue may form at the runner with the data waiting to be entered by a data clerk, who cannot work fast enough. Upon completion of the data entry, the clerk assigns the file to one of six districts around the state Each district has a group of specific dealers determined by geographic distribution. and the resulting application now electronic in form is transmitted via WLAN to one of FSB's regional offices.

Phase 3: A loan officer in that region than processes the loan, correcting any errors in the application and then makes a final decision.
Target Processing:The goal is that data entry should take no longer than six minutes. The goal was also set that there should be no greater than 5 percent errors. Errors are corrected at the region and add to the region's processing time.and he loan officer should complete this function within 20 minutes.

Phase 4: The decision is faxed back to the dealer directly from the regional office.

Target Processing Times:

The target average values that Todd suggests should be achieved:

- -Turnaround time: 30 minutes
- -Loan processing: 20 minutes
- -Data entry: 6 minutes
- -Error rate: 5%

Team Members

Team Leader:	Ali Mowazi - CS 4th Year
Conceptual Team:	Rahman Suleiman - SYS Masters
	Peter Wu - SYS Masters
Simulation Team:	Su Zhang - CS 3rd Year
	Ryan Warnock - CS 4th Year

2. Project Goals

The goal of the project is to model a system that gives similar results to the measured results in the real system. To then solve the problem of extended turnaround time we will address the issues of correcting processing times and lowering the error rates. To do this we will vary the number of loan officers also putting into consideration the time it takes the loan officers to review the loan application by lowering the time and data clerks through experimentation until we can decrease the speed of the turnaround time.

Parameters

- RG.HQDataClerks.numClerks: number of data entry clerks at HQ Boise.
- numOfficersDistricts: Sequence of 6 values that specify number of Loan Officers at each district <COEUR D'ALENE,LEWISTON,BOISE,SHOSHONE,POCATELLO,RIGBY> where Couer d'Alene,Boise… is an identifier of each member of the set.
- pcntError: percent error
- dataEntryAvgTime: service time for data entry clerks.
- districtProcessingAvgTime: service time for loan officers

Experimentation

Study: Steady-State Study
Observation Interval:

- Time units are minutes
- Interval of observation cannot be predicted as steady-state must be reached

Base Case: The number of loan officers in each field is <4.2,6,3,2,2>, the numbers correspond to districts 1-6 respectively and 8 data clerks at HQ Boise. Percent error is 10%, value of district Processing Avgtime <20 or 18 or 16>, Service time for data entry is average 9.5 minutes.

Alternate Cases:

Each set of experiments will be conducted with the following parameters to determine the number of clerks and loan officers needed to meet the target turnaround time (24 minutes).

Experimental Trial No.	Data Clerk ST (Min)	Office ST (Min)	Error Rate(%)
1	6	20	5
2	5	18	5
3	4	16	5

Experiment Step 1: Set RG.HQDataClerks.numClerks to 30. This ensures no bottleneck at data entry. Increase loan officers by one at each district that does not meet the loan processing time. Repeat until all districts meet desired average turnaround time.

Step 2: Keep the loan officers from Step 1 and set RQ.HQDataClerks.numClerks = 8. Increase this parameter until desered average turnaround time is met.

Output

- avgApprovalTurnaround [*regional office*] - Average time taken for loan to be approved (including queue and service time) for each regional office
- avgTotalTurnaround - Average total turnaround time for all loan applications

3. ABCmod Conceptual Model

High Level Conceptual Model

Assumptions

- FSB operations run for 20 days/month and 8 hours a day.

- Assume service time (for data entry/loan approvals) are similar to random uniform distributions with a range +/- 25% of the average.

Simplifications

- There will never be a queue for fax machines as there is assumed to be sufficient capacity. (For instance, an industrial grade fax machine could receive and print many applications at once).

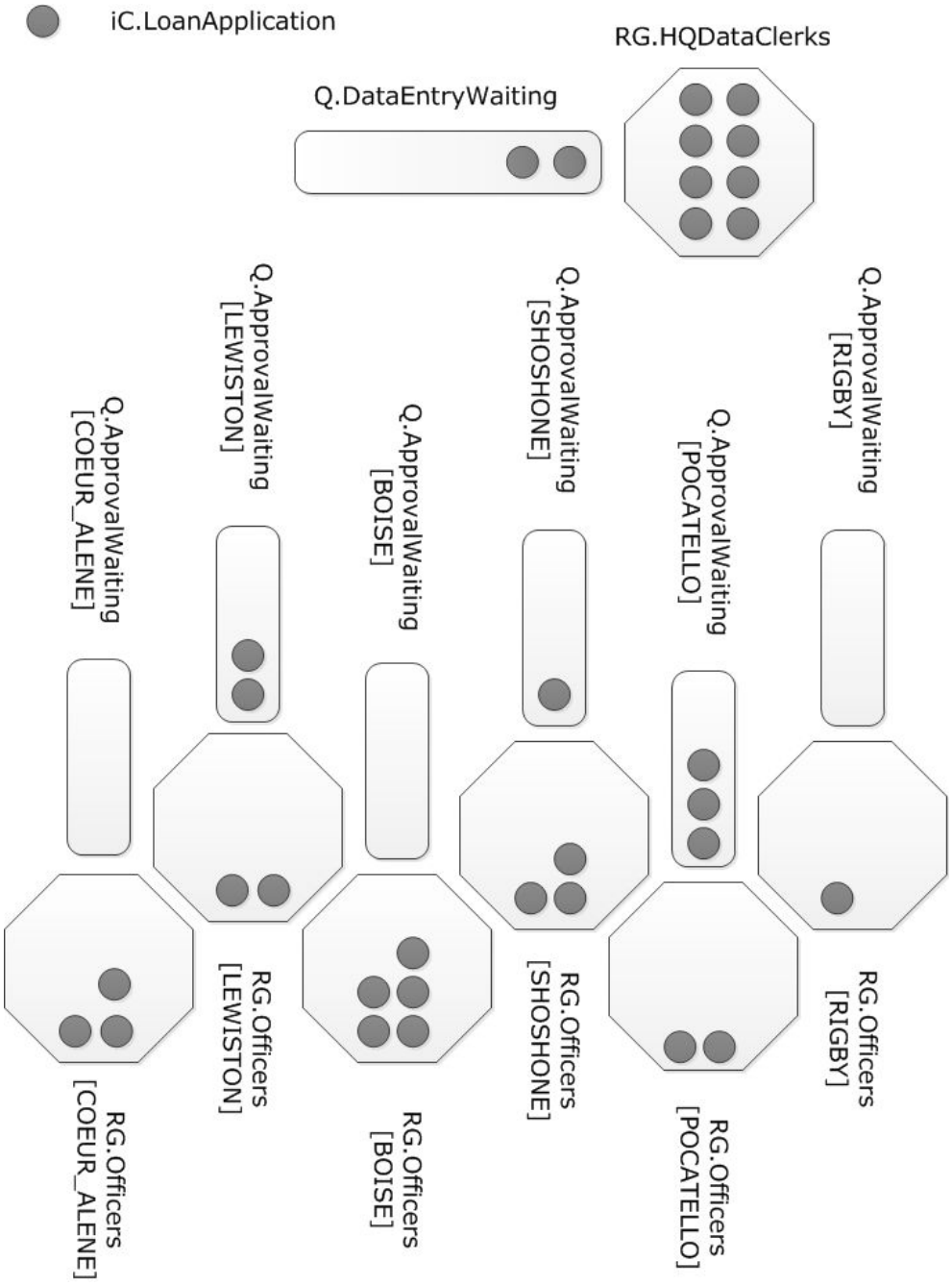
- It is assumed that WLAN transmission and runners take insignificant (less than the discrete time increments considered) amounts of time. WLAN transmissions are fast, and the data entry clerks may be located in very close proximity to the fax machines (i.e. their behaviour could be analogous to a worker removing parts from a conveyer belt).

- Fax machines are assumed as a constant time task rather than stochastic time as the variance in transmission time is likely to be minimal.

- Fax machines will not need to be modelled since they have no queue and work in constant time.

- Individual data entry clerks and loan officers are not modelled, as their behaviours are captured within the group models

Structural View



Entities:

iC.LoanApplication -- A consumer entity representing the stream of loan applications being sent to FSB for processing. This is a transient entity with *scope* = *Class*.

Q.DataEntryWaiting -- Standard queue entity that represents the number of applications awaiting data entry at HQ.

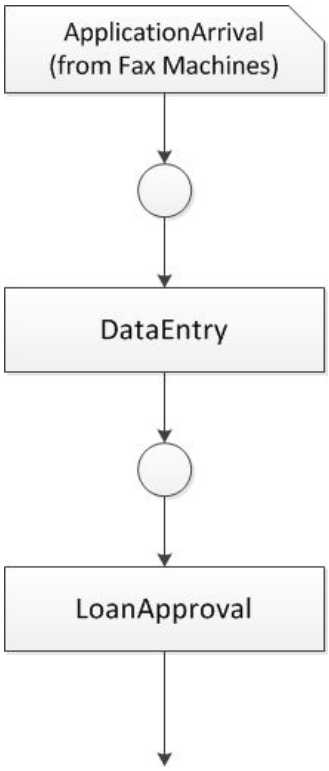
RG.HQDataClerks -- A resource-group that represents the data entry workers at HQ.

Q.ApprovalWaiting[*regional office*] -- Standard queue entity that represents the number of applications awaiting data entry at HQ.This entity has *scope = Set[6]* representing the office in each region.

RG.Officers[*regional office*] -- A resource-group that represents the regional loan approval officers. This entity has *scope = Set[6]* representing the office in each region.

Behavioural View

The following behavioural diagram describes the lifecycle of a Loan Application under modelling consideration:



Actions:

ApplicationArrival -- Scheduled action generates input stream of application arrivals initialized with the Origin attribute which determines which district office processes the application. This

represents the FSB Headquarters receiving a loan application at their fax bank and the runners placing it immediately in queue for data entry.

Activities:

DataEntry -- This activity represents the data entry clerks transcribing the application.

LoanApproval -- This activity represents the processing tasks in the various districts after the data input is completed (errors are corrected and application is reviewed). For an application that goes through this activity, there is a set probability that the approval officer discovers a transcription error and will have to spend additional time to correct it.

Input

Endogenous Input (Semi-Independent)		
Variable Names	**Description**	**Values**
uDataEntryTime	The time it takes to input the data by the data entry clerk.	RVP.uDataEntryTime()
uDistrictProcessingTime	The time it take for a district officer to process the loan application. The base mean service time is a parameter and an error will randomly be detected for a set percentage of processed applications--this will incurr a 6 minute time penalty (i.e. shifting the distribution mean right by 6 minutes).	RVP.uDistrictProcessingTime()
iC.LoanApplication.uOrigin	Which region the loan app originated from.	RVP.uLoanAppOrigin()

Exogenous Input (Entity Stream)			
Variable Name	**Description**	**Domain Sequence**	**Range Sequence**
uLoanAppArrival	This represent when the loan application is received at the HQ fax machine bank.	RVP.DuLoanAppArrival()	N/A - 1 loan application arrives at each arrival time

Detailed Conceptual Model

Structural Components

CONSTANT		
NAME	DESCRIPTION	VALUE
REG_PCNT_APPS	This is the average percent of applications that originates from each district	<17.7, 9.6, 41, 10.6, 9.9, 11.2> representing the percentage of applications originating from COEUR_ALENE, LEWISTON, BOISE, SHOSHONE, POCATELLO and RIGBY
COEUR_ALENE	Regional index	1
LEWISTON	Regional index	2
BOISE	Regional index	3
SHOSHONE	Regional index	4
POCATELLO	Regional index	5
RIGBY	Regional index	6

PARAMETERS		
NAME	DESCRIPTION	VALUE
RG.HQDataClerks.numClerks	Number of data entry clerks available	8, 8+ if necessary (extent of increase dependant on results after each trial)
numOfficersDistricts	Used to initialize RG.Officers[*regional office*].numOfficers, which are	< ≥4, ≥2, ≥6, ≥3, ≥2, ≥2> representing the number of loan officers from

	the number of loan officers available in each district	COEUR_ALENE, LEWISTON, BOISE, SHOSHONE, POCATELLO and RIGBY (extent of increase dependant on results after each trial)
pcntError	Percentage of applications with error	10% or 5%
dataEntryAvgTime	Average time to complete the DataEntry activity	6 or 5 or 4 min
districtProcessingAvgTime	Average time to complete the LoanApproval activity	20 or 18 or 16 min

CONSUMER CLASS: LoanApplication The consumer entity structure gives a representation of the loan application from various dealers within the Districts.	
ATTRIBUTES	DESCRIPTION
uOrigin	Tracks which district the loan application came from (COEUR_ALENE, LEWISTON, BOISE, SHOSHONE, POCATELLO or RIGBY)
startTime	This is the time value when the loan application is received at the HQ fax machines and a runner picks it up
endDataEntryTime	Time stamp of when the data entry completes for the loan application

RESOURCE GROUP UNARY: HQDataClerks This resource group represents the data entry clerks.	
ATTRIBUTES	DESCRIPTION
numClerks	This represents the number of data clerks
list	The group of applications being “serviced” by data clerks

n	The number of applications being processed (in list) which also gives the number of occupied data clerks

QUEUE UNARY: DataEntryWaiting This queue represents the applications waiting for data entry by clerks.	
ATTRIBUTES	DESCRIPTION
list	List of loan applications waiting to be “serviced” by data clerks
n	Length of the list

RESOURCE GROUP SET [6]: RG.Officers This resource group that represents the loan approval officers for each region. (Identifiers are: COEUR_ALENE, LEWISTON, BOISE, SHOSHONE, POCATELLO and RIGBY)	
ATTRIBUTES	DESCRIPTION
numOfficers	This represents the number of loan officers
list	The group of applications being “serviced” by loan officers
n	The number of applications being processed (in list) which also gives the number of occupied loan officers

QUEUE SET [6]: Q.ApprovalWaiting This queue represents the applications waiting on loan approval for each region. (Identifiers are: COEUR_ALENE, LEWISTON, BOISE, SHOSHONE, POCATELLO and RIGBY)	
ATTRIBUTES	DESCRIPTION
list	List of loan applications waiting to be “serviced” by loan officers
n	Length of the list

Behavioural components

Time units: Minutes

Observation interval: Steady state study determined during experimentation

Output

SAMPLE SEQUENCES	
ATTRIBUTES	DESCRIPTION
PHI[approvalTurnaround [*regional office*]]	(This is actually 6 separate sequences, one for each region) Each pair in the sequence is of the form (tk, yk) where yk is the time it takes for an application to queue and get approved (or rejected at the regional office, and tk is when the application is received. Computed as t - iC.LoanApplication.endDataEntryTime
PHI[totalTurnaround]	Each pair in this sequence is of the form (tk, yk) where yk is the time from when the application is received to when approval is granted (or rejected), and tk is when the application is received. Computed as t - iC.LoanApplication.startTime

DERIVED SCALAR OUTPUT VARIABLES (DSOVs)			
ATTRIBUTES	DESCRIPTION	OUTPUT SEQUENCE NAME	OPERATION
avgApprovalTurnaround [*regional office*]	(This is actually 6 separate DSOVs, one for each region) Average time taken for loan to be approved (including queue and service time)	PHI[approvalTurnaround [*regional office*]]	AVG
avgTotalTurnaround	Average total turnaround time for all loan applications	PHI[totalTurnaround]	AVG

Input Constructs

RANDOM VARIATE PROCEDURES		
NAME	DESCRIPTION	VALUE
RVP.DuLoanAppArrival()	Provides the values of the arrival times of the loan applications.	First arrival at t = 0 and interarrival given by exponential distribution EXP(X) where X is 3.8 minutes
RVP.uDataEntryTime()	Provides a value for the time it takes for the data entry clerks to enter an application.	Uniform distribution given by UNIFORM(Min, Max) where min = dataEntryAvgTime * 0.75 and max = dataEntryAvgTime * 1.25
RVP.uDistrictProcessingTime()	Provides a value for the time it takes for the loan approval officers to complete application processing.	Uniform distribution given by UNIFORM(Min, Max) where min and max may take on the following values: <u>Without Error:</u> min = districtProcessingAvgTime * 0.75 and max = districtProcessingAvgTime * 1.25 <u>With Error:</u> min = (6 + districtProcessingAvgTime) * 0.75 and max = (6 + districtProcessingAvgTime) * 1.25 Where the proportion of applications with error are given by pcntError and without error = (100% - pcntError)

RVP.uLoanAppOrigin()	Provides the origin of where an application comes from.	Proportion of applications originating from COEUR_ALENE, LEWISTON, BOISE, SHOSHONE, POCATELLO and RIGBY as given by regPcntApps

USER DEFINED PROCEDURES	
NAME	DESCRIPTION

LoanApplicationAvailable	*(Checks each district to see if there is a free space and an application in queue for that district)*
	
	districtID = NONE
	
	FOR i = (1..numOfficersDistricts.length)
	IF (RG.Officers[i].n < RG.Officers[i].numOfficers AND
	Q.ApprovalWaiting[i] > 0)
	districtID ← i
	BREAK
	END FOR
	
	RETURN districtID

Behavioural Constructs

ACTION: Initialize The Input Entity Stream of arriving loan applications	
TIMESEQUENCE	<0>
EVENT SCS	*(Initializes certain entities with initialization requirements)* FOR i = (1..numOfficersDistricts.length) RG.Officers[i].numOfficers ← numOfficersDistricts[i] END FOR

ACTION: ApplicationArrival The Input Entity Stream of arriving loan applications	
TIMESEQUENCE	RVP.DuLoanAppArrival()
EVENT SCS	*(Standard initialization)* iC.LoanApplication ← SP.Derive(LoanApplication) iC.LoanApplication.uOrigin ← RVP.uLoanAppOrigin() iC.LoanApplication.startTime ← t SP.InsertQue(Q.DataEntryWaiting, iC.LoanApplication)

ACTIVITY: DataEntry Data clerks transcribe the loan application data	
PRECONDITION	*(Checks if there are any free clerks and if there are any applications in queue for data entry)* (RG.HQDataClerks.n < RG.HQDataClerks.numClerks) AND (Q.DataEntryWaiting.n != 0)
EVENT SCS	*(Moves the application to the data clerks group)* iC.LoanApplication ← SP.RemoveQue(Q.DataEntryWaiting) SP.InsertGroup (RG.HQDataClerks, iC.LoanApplication)
DURATION	RVP.uDataEntryTime()
EVENT SCS	*(Timestamps when this process finishes, moves application into a regional office based on its origin attribute)* iC.LoanApplication.dataEntryEndTime ← t SP.RemoveGroup(RG.HQDataClerks, iC.LoanApplication) SP.InsertQue(Q.ApprovalWaiting[iC.LoanApplication.uOrigin], iC.LoanApplication)

ACTIVITY: LoanApproval

Loan officers review the loan application data and correct errors if necessary	
PRECONDITION	*(Checks if any regional office has free officers, and then checks to see if any applications are in queue for that regional office)* districtID ← LoanApplicationAvailable() (districtID != NONE)
EVENT SCS	*(Moves the application to the appropriate office)* iC.LoanApplication ← SP.RemoveQue(Q.ApprovalWaiting[districtID]) SP.InsertGroup (RG.Officers[districtID], iC.LoanApplication)
DURATION	RVP.uDistrictProcessingTime()
EVENT SCS	*(Updates output sample sequences and removes applications)* SP.RemoveGroup(RG.Officers[*regional office*], iC.LoanApplication) SP.Put(PHI[regApprovalTurnaround[*regional office*]], t - iC.LoanApplication.dataEntryTime) SP.Put(PHI[totalTurnaround], t - iC.LoanApplication.startTime) SP.Leave(iC.LoanApplication)

Design of Validation Experimentation

Given the simplicity of the model, validations will be provided through trace log. The following is expected to be observed through the model runs:

- Some application passing through all the appropriate entities
- Applications ending up in the correct regional office based on their origins
- Queues forming when resource entities are fully occupied
- Timestamps progressing in a reasonable fashion (based on experimenter judgment and relevant parameter settings)

Trace Logging

The state of the simulation model is monitored by tracking the state of the two queues and resource groups. An example output may look like this:

```
Timestamp:XXX
================
Q.DataEntryWaiting.n: X |  RG.HQDataClerks.n: X
Q.ApprovalWaiting[COEUR_ALENE].n: X | RG.Officers[COEUR_ALENE].n: X
Q.ApprovalWaiting[LEWISTON].n: X | RG.Officers[LEWISTON].n: X
```

```
Q.ApprovalWaiting[BOISE].n: X | RG.Officers[BOISE].n: X
Q.ApprovalWaiting[SHOSHONE].n: X | RG.Officers[SHOSHONE].n: X
Q.ApprovalWaiting[POCATELLO].n: X | RG.Officers[POCATELLO].n: X
Q.ApprovalWaiting[RIGBY].n: X | RG.Officers[RIGBY].n: X

Details:
Q.DataEntryWaiting.list:
[XXX,XXX,..]
RG.HQDataClerks.list:
[YYY,ZZZ,..]
Origins - YYY: YYY.uOrigin | ZZZ: ZZZ.uOrigin | etc..
Q.ApprovalWaiting[COEUR_ALENE].list:
[XXX,XXX,..]
RG.Officers[COEUR_ALENE].list:
[XXX,XXX,..]
Q.ApprovalWaiting[LEWISTON].list:
[XXX,XXX,..]
RG.Officers[LEWISTON].list:
[XXX,XXX,..]
Q.ApprovalWaiting[BOISE].list:
[XXX,XXX,..]
RG.Officers[BOISE].list:
[XXX,XXX,..]
Q.ApprovalWaiting[SHOSHONE].list:
[XXX,XXX,..]
RG.Officers[SHOSHONE].list:
[XXX,XXX,..]
Q.ApprovalWaiting[POCATELLO].list:
[XXX,XXX,..]
RG.Officers[POCATELLO].list:
[XXX,XXX]
Q.ApprovalWaiting[RIGBY].list:
[XXX,XXX,..]
RG.Officers[RIGBY].list:
[XXX,XXX]
================
```

Where, XXX, YYY, ZZZ, etc.. are replaced with the attribute values accordingly.

4. Simulation Model

Design of Simulation Model and Program

The simulation model is implemented in the class FSB (an extension of the AOSimulation model) and a number of others classes are used to implement the various constructs from the ABCmod conceptual model.
The following table shows the various ABCmod entity structures and their corresponding Java classes and how objects instantiated from these classes are referenced by the FSB class.

Entity Structures		
ABCmod Construct	Java Class	Object References
iC.LoanApplication	LoanApplication Notes: The enumerated data type uOrigin keeps track of the regional office that the application corresponds to.	icLoanApplication
RG.HQDataClerks	HQDataClerks Notes: -The list of LoanApplication objects is maintained in a HashSet object. The methods add and remove are used to move applications from this set. -The attribute n represents the number of applications being entered and thus also the number of busy data clerks.	FSB.rgHQDataClerks
RG.Officers	Officers Notes: -The list of LoanApplication objects is maintained in a HashSet object. The methods add and remove are used to move applications from this set.	FSB.rgOfficers

	-The attribute n represents the number of applications being processed and thus also the number of busy loan officers.	
Q.DataEntryWaiting	ArrayList Notes: -The LoanApplications are stored in the ArrayList which acts as a queue. -The methods of the ArrayList class are used to add and remove elements from the list.	qDataEntryWaiting
Q.ApprovalWaiting	ArrayList Notes: -A nested ArrayList is used to have six ArrayLists, one for each district. Each of these inner ArrayLists works as the Q.DataEntryWaiting ArrayList does.	qApprovalSet

The following table provides mapping between the conceptual model Action and Activities classes and the corresponding Java classes.

Actions/Activities	
ABCmod Constructs	Java Classes
LoanApproval	LoanApproval
DataEntry	DataEntry
ApplicationArrival	AppArrival

Other classes that make up the FSB/AOSimulation model include:
-Constants: Contains constants including the loan arrival distribution weights among districts.
-Output: Contains the two output sample sequences, phiICapprovalTurnAround and phiICtotalTurnAround. Includes methods for retrieving these turnaround times.

-RVPs: Contains the java methods used to implement the RVPs found in the conceptual model.
-Seeds: The class used to pass randomly generated values used by the RVPs.
-UDPs: Contains the method getID that returns the origin ID of a loan application.

Annex A – Data Modelling

Data modelling information:

District	Head Quarters	Applications	Average Time	Number of Loan Officers
1	Coeur d'Alene	2655	37.00	4
2	Lewiston	1440	37.00	2
3	Boise	6150	58.76	6
4	Shoshone	1590	37.01	3
5	Pocatello	1485	37.22	2
6	Rigby	1680	51.07	2

Total applications: 15000
Total time: 712804.2 minutes
Weighted average time: 46.07 minutes

FSB currently has the following average rates for each loan application:

-Data entry: 9.5 minutes
-Error rate: 10%
-Error processing time: 6 minutes

Mean Arrival Time:
Assume operations are 20 days/month with 8 hours/day:

6 months x 20 [days/month] x 24 [hours/day] x 60 [min/hour] = 57600 minutes

15000 applications in 6 months of data:

57600 min / 15000 applications ~= 3.8 minutes/application (avg)

Regional Percentages:

2655 applications from Coeur d'Alene:

2655 / 15000 = 0.177 = 17.7%

1440 applications from Lewiston:

1440 / 15000 = 0.096 = 9.6%

6150 applications from Coeur d'Alene:

6150 / 15000 = 0.410 = 41%

1590 applications from Coeur d'Alene:

1590 / 15000 = 0.106 = 10.6%

1485 applications from Coeur d'Alene:

1485 / 15000 = 0.099 = 9.9%

1680 applications from Coeur d'Alene:

1680 / 15000 = 0.112 = 11.2%