ORM
TECHNOLOGIES
Optimized Thinking.

PRESENTED TO:

DR. RICHARD BARR

DR. THOMAS SIEMS

EMIS FACULTY AND STUDENTS

ORM TECHNOLOGIES

PRESENTED BY:

ALEXANDRIA FARRAR

ASHLEY HALL

NEIMY SARMIENTO

SOUTHERN METHODIST UNIVERSITY

EMIS 4395: SENIOR DESIGN

SPRING 2014

TABLE OF CONTENTS

MANAGEMENT SUMMARY 2
BACKGROUND AND DESCRIPTION OF THE PROBLEM SITUATION 3
ANALYSIS OF THE SITUATION 4
TECHNICAL DESCRIPTION OF THE MODEL 7
ANALYSIS AND MANAGERIAL INTERPRETATION 14
CONCLUSIONS AND CRITIQUE 15

WEEK ONE PROGRESS REPORT 16
WEEK TWO PROGRESS REPORT 18
WEEK THREE PROGRESS REPORT 19
WEEK FOUR PROGRESS REPORT 20
WEEKS FIVE AND SIX PROGRESS REPORT 21
WEEK SEVEN PROGRESS REPORT 22
WEEK EIGHT PROGRESS REPORT 23
WEEK NINE PROGRESS REPORT 24
WEEK TEN PROGRESS REPORT 27
WEEK ELEVEN PROGRESS REPORT 29
WEEK TWELVE PROGRESS REPORT 31
WEEK THIRTEEN PROGRESS REPORT 33

MANAGEMENT SUMMARY

ORM Technologies currently has a call center client looking to make their call center more efficient. Currently, ORM Technologies has a functioning system that uses web based inputs that allowing the customer to input data and generate optimized resource requirements. The system runs an Erlang C profile for every agent level in the pre-model. From there, the optimization selects the minimum headcount required by 30 minute intervals by day, week, month and year. Some models are 12 months, but most are 5 years.

Our team had the task of using Erlang B probability to dynamically estimate and model the most cost effective number of trunks, or phone channels, needed to service a live call center by trunk type. Some of the constraints associated with the project and model include:

1. Its language; it must be written in GAMS, General Algebraic Modeling System. Our team has no prior experience with GAMS.
2. Erlang B must be used as it can be integrated into the client's current model.
3. 250-trunk maximum
4. Time; the project has a four month time constraint

We communicated our progress with ORM Technologies and our project advisor, Dr. Richard Barr. At their advice, we started by researching GAMS and Erlang B and worked with a data set provided by ORM. The data was segmented by 30-minute time intervals from 7AM to 6:30PM, Monday through Friday for 60 months. Initially, we found the peak call volume in the given data and used that value as the input for an online Erlang B calculator. Next, we compared the online calculator's output to an Excel Erlang B model; the results were consistent.

Now, we need to put our Excel model into GAMS. Due to our novice knowledge of GAMS, our initial model was written in Python. Once familiarity was gained, our code was translated into GAMS.

After presenting our progress to ORM, we were instructed to integrate a cost analysis for three separate trunk types into our GAMS model. We found the annual trunk cost to be $7,140 with quarterly peak trunk volumes at March, June, September, and December resembling a cosine wave.

BACKGROUND AND DESCRIPTION OF THE PROBLEM SITUATION

ORM Technologies currently has a client with a call center that is looking to make their call center more efficient. Currently, ORM Technologies has a functioning system that uses web based inputs that allowing the customer to input data and generate optimized resource requirements. The system runs an Erlang C profile for every agent level in the pre-model. From there, the optimization selects the minimum headcount required by 30 minute intervals by day, week, month and year. Some models are 12 months, but most are 5 years.

They typically have 1-5 call types and 1-10 agent types that have different rules and restrictions. The current blending approach recalculates the volumes by call type on available capacity. Secondly, it tests several levels of transfer of the call volume, which creates 5 or 6 times more choices to select from for blending. Upon completion of this model, the results can then be passed to a Shift Scheduling Manager that determines the optimal shift balance that will to satisfy the blended or unblended requirements.

The current system is a web-based client interface that runs at Amazon Web Services where the client data is captured and results presented. The Amazon server formats the new inputs into a package. This package file is then passed to ORM’s server in Dallas where the optimization models reside (GAMS-based model) and the run is executed. From there, ORM passes the results back to the Amazon server where a number of presentation calculations are performed and then rendered to the client.

The requirement we will complete is to add Erlang B provisioning calculations and integrate these calculations into their current framework for dynamic modeling. Erlang B is a call-center modeling probability formula that can be used to calculate any one of the following three factors if you know or predict the other two:

1. Busy Hour Traffic (BHT): the number of hours of call traffic during the busiest hour of operation

2. Blocking: the percentage of calls that are blocked because not enough lines are available

3. Lines: the number of lines in a trunk group.

ANALYSIS OF THE SITUATION

As discussed, our primary goal was to build a model using Erlang B probabilities to determine the optimal number of trunks needed to service the inbound call volume into the Public Branch Exchange (PBX), at any given time. Once the optimal number of trunks was obtained, we next calculated the most cost effective solution using GAMS. In order to build the model we first had to clearly understand how Erlang B probabilities work, what we were given and what constraints we were facing.

<u>Erlang B</u>

Erlang B is a probability formula used to calculate number of optimal trunks needed to carry incoming calls. The equation is as follows:

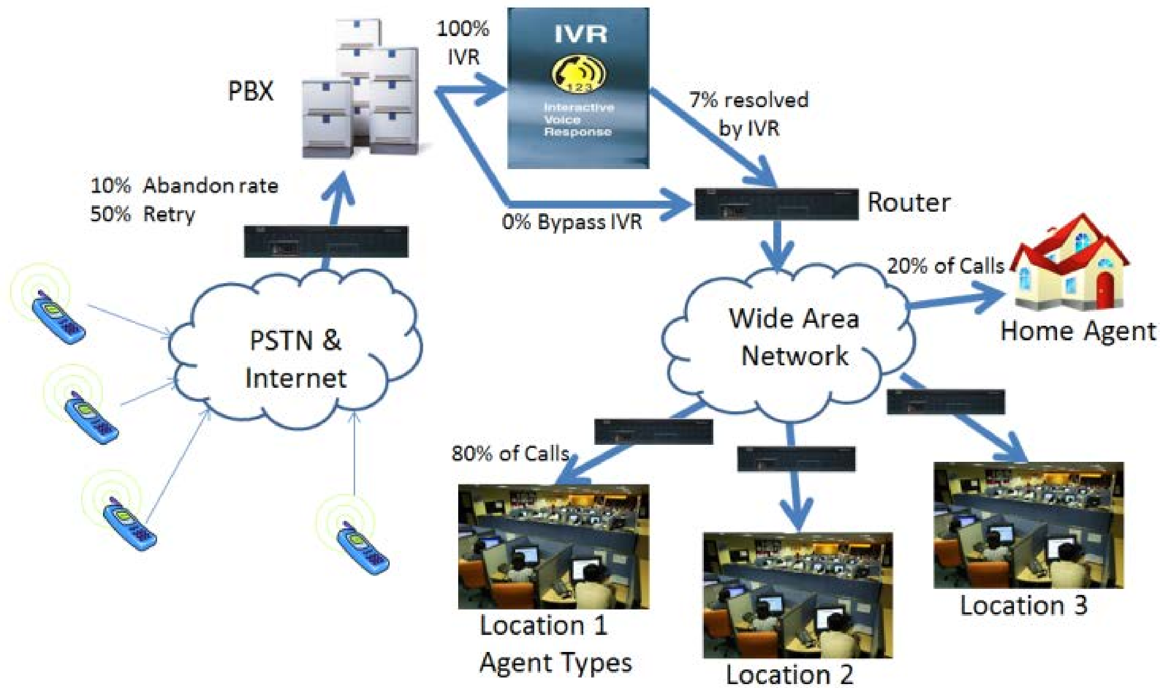
$$P_b = \frac{\frac{E^m}{m!}}{\sum_{i=0}^{m} \frac{E^i}{i!}}$$

$\boldsymbol{P_b}$**:** grade of service; the acceptable level of call blocking.
E = **λ*h:** traffic intensity in Erlangs
λ = arrival rate
h = average handling time
m = number of trunks

For the purpose of our model, we chose to use a grade of service of 1%. Anything higher would result in unacceptable service level objective, and anything lower would be too costly. Due to the complexity of the equation, we could not extract the variable *m* and find the optimal number of trunks; rather we found the number of trunks that would satisfy our desired grade of service through an iterative model.

Once we understood Erlang B, we had to adapt our data to satisfy the equation. Our data set included the number of inbound calls for any given 30 minute time interval over 60 months, Monday through Friday. From this we needed to calculate *E*, the traffic intensity.

The issue we encountered in this was that the data we were given only included the calls going into the Wide Area Network (WAN) as demonstrated in the flow chart below. In order to truly obtain the traffic intensity, we needed to know the total call volume going into the PBX. This required a recursive algorithm due to the abandon and retry rate that occur when calls enter the Public Switch Telephony Network (PSTN). From this call volume we calculated the traffic intensity; it is the product of the arrival rate of this volume and its average handling time.



Because our model needs to be flexible and easily adaptable in order to work for a variety of ORM clients, we understood the need to allow for easy data import. In addition to this, some call centers have a variety of call types; this would also have to be worked into the model. The data set used to build our model consisted of three different call types. This adds another level of complexity when finding the traffic intensity as the average handling time differs for different call types.

Once the traffic intensity is determined, the goal is to be able to directly import this data into our GAMS model. The model will find *m*, the number of trunks needed to satisfy the required grade of service, and determine the least costly combination of trunk types that should be purchased.

In summary we took the following steps:

1) Understand Erlang B
2) Backtrack our given volumes to reflect the gross traffic volume entering the PBX
3) Apply recursive algorithm to satisfy the abandon and retry rates
4) Design model for easy data import
5) Design model for varying call types
6) Attain *E*, the traffic intensity
7) Calculate the least number of trunks needed using Erlang B

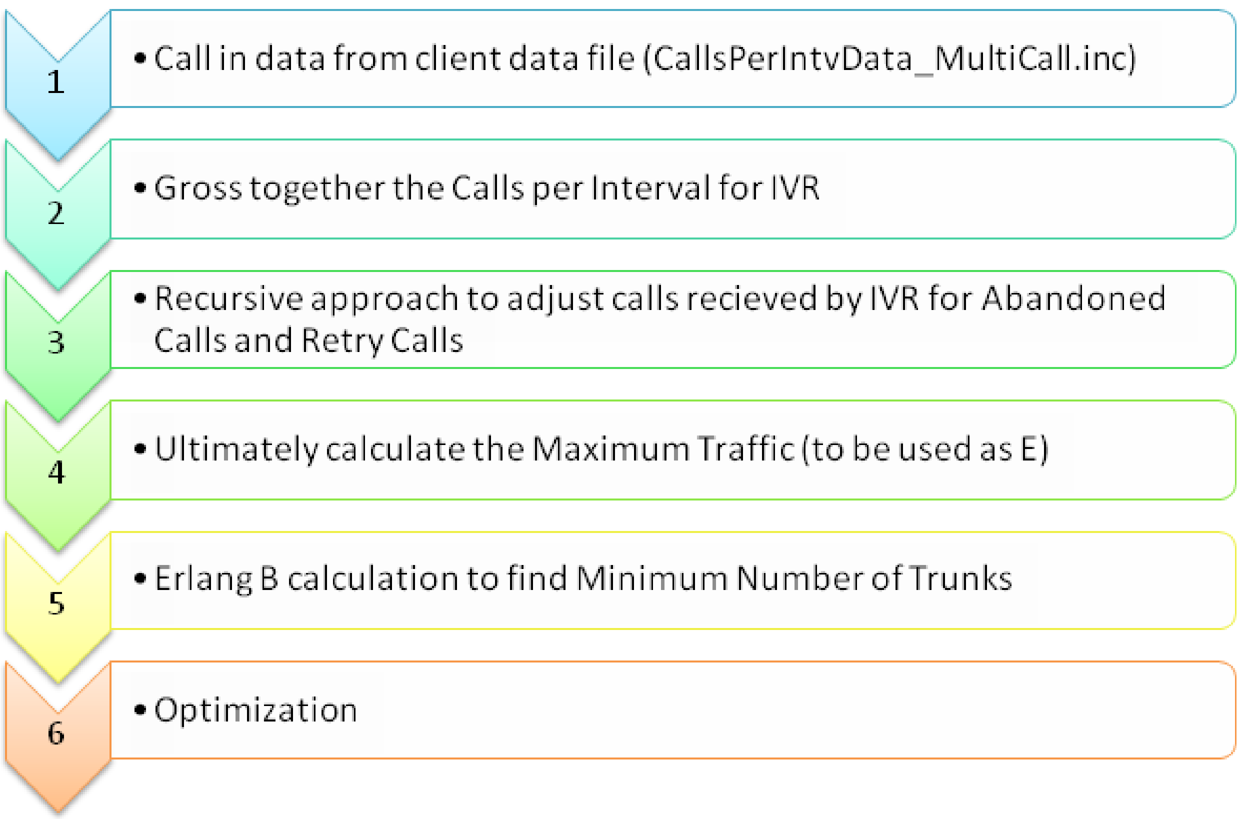
8) Calculate the minimum cost combination of trunk types

The most problematic steps were understanding Erlang B probabilities and completing the necessary steps to attain *E*, the traffic intensity.

TECHNICAL DESCRIPTION OF THE MODEL

The optimization model used in order to solve the problem was General Algebraic Modeling System (GAMS). We used this modeling program not only to perform the optimization portion of the modeling, but also the mathematical calculations. The mathematical calculations that we needed to perform included finding the total number of calls to the Public Branch Exchange (PBX), starting with the number of calls that reached the Wide Area Network (WAN). The GAMS SMU_ErlangB_Calculator.gms has multiple distinct stages.

A flowchart of the optimization model is shown in Figure 1 below. Following the flowchart is an explanation of all the parts.



The first portion included in this model reads in all the calls per interval from the data set CallsPerIntvData_MultiCall.inc which is provided by the client. Since this model is able to handle multiple call types, this model will also be able to handle data sets with only single call types as well. Our model set all the scalar values; the most noteworthy scalars being: the Erlang B

Service Level (0.01), IVR Percentage (0.07), Call Abandon Percentage (0.1), and Call Retry Percentage (0.5). A portion of the data file is shown below in order to represent the different

1 • Call in data from client data file (CallsPerIntvData_MultiCall.inc)

call types

```
Parameter CallsPerInterval(CallType,month,Interval,Days)/
"#9: Walkins"."1"."08:30"."1" = 2.50
"#9: Walkins"."1"."08:30"."2" = 2.19
"#9: Walkins"."1"."08:30"."3" = 1.88
"#9: Walkins"."1"."08:30"."4" = 2.00
"#9: Walkins"."12"."16:30"."5" = 0.65
"#10: BackOffice Work"."1"."08:30"."1" = 2.32
"#10: BackOffice Work"."1"."08:30"."2" = 2.03
"#10: BackOffice Work"."1"."08:30"."3" = 1.74
"#10: BackOffice Work"."12"."16:30"."4" = 0.60
"#10: BackOffice Wo   '."12"."16:30"."5" = 0.60
"#11: Front Office Calls"."1"."08:30"."1" = 48.68
"#11: Front Office Calls"."1"."08:30"."2" = 42.60
"#11: Front Office Calls"."1"."08:30"."3" = 36.54
```

The second portion of the model grosses together the calls per interval for the IVR calls handled. The model calculates the actual calls by call type, per interval, by day and month. It then takes the calls by call type for the month and converts the calls into weekly values. Then the model allocates the weekly value to a total by day which then allocates the daily balance to each interval. This portion then displays the:

1. Calls per Interval
2. Total Calls per Interval
3. Maximum Calls per Interval
4. Maximum of all Call Types per Interval
5. Maximum Traffic from all the Intervals.

The variables mentioned above are used to determine the IVR Calls per Interval, the Total Calls per Interval, the IVR Call Percentage, and the Maximum IVR Calls. The most important variable from these is the IVR Calls per Interval. The equation used to calculate the IVR calls is the following:

$$IVRCallsPerIntv = \frac{CallsPerInterval(CallType, Month, Interval, Days}{1 - IVRPct}$$

GAMS calculates the gross traffic value based on calls handled by the IVR and also for calls abandoned and call retried. The first stage is to add back the calls that the IVR handled without passing them to the agents. We had to remember that the agent calls from above are the net calls by call type received by the agents after the IVR or any other device has worked with the call. Given the IVR resolves for seven percent of the calls it handles, we needed to gross up the agent net calls in order to recognize that the inbound to the IVR is higher.

2

- Gross together the Calls per Interval for IVR

```
----    3263 PARAMETER IVRCallsPerIntv

INDEX 1 = #9: Walkins

                  1           2           3           4           5

1 .08:30    2.68817     2.35484     2.02151     2.15054     2.68817
1 .09:00    3.89247     3.02151     3.02151     3.02151     3.90323

12.16:30    0.88172     0.44086     0.70968     0.69892     0.69892

INDEX 1 = #10: BackOffice Work

                  1           2           3           4           5

1 .08:30    2.49462     2.18280     1.87097     2.00000     2.49462
1 .09:00    3.62366     2.80645     2.80645     2.81720     3.62366
12.16:30    0.81720     0.40860     0.65591     0.64516     0.64516

INDEX 1 = #11: Front Office Calls

                  1           2           3           4           5

1 .08:30   52.34409    45.80645    39.29032    41.89247    52.31183
1 .09:00   75.84946    58.82796    58.87097    58.88172    75.87097
```

The third portion of the model adjusted the calls received by the IVR for abandoned calls and retry calls with a recursive approach. We do this in order to deduct the net calls presented to the PBX from the network. The values for Abandon Calls, Abandon Retry, and Calls to the PBX all had to be fixed for the first position of the retry loop. Then the loop ran from position two to twenty, or as many as necessary based on the limitations. The very first Abandon Calls number is the number of calls the IVR is handling multiplied by the Call Abandon Rate. This is only for the first calculation and then the abandon calls are based on the people who retried and then abandoned again. The second loop calculates the Net Calls to the PBX by call type. All this work has been done by interval and call type.

A snapshot of the recursive loop is shown in Figure 2 below:

3

- Recursive approach to adjust calls recieved by IVR for Abandoned Calls and Retry Calls

```
display IVRCallsPerIntv,TotIVRCalls,IVRCallPct,MaxIVRCalls;

*-----------------------------------------------------------------------------------------
Set RetryLoop/R1*R20/;
Parameter AbandonCalls(RetryLoop,CallType,month,Interval,Days),AbandonRetry(RetryLoop,CallType,month,Interval,Days),
          Calls2PBX(RetryLoop,CallType,month,Interval,Days);

* fix values for retryloop position 1 then the loop handles position 2 - 20 or however many are necessary based on the limitations.
AbandonCalls(RetryLoop,CallType,month,Interval,Days)$(RetryLoop.pos=1)
                                 = IVRCallsPerIntv(CallType,Month,Interval,Days)*CallAbandonPct;
AbandonRetry(RetryLoop,CallType,month,Interval,Days)$(RetryLoop.pos=1)
                                 = AbandonCalls(RetryLoop,CallType,month,Interval,Days)*CallRetryPct;
Calls2PBX(RetryLoop,CallType,Month,Interval,Days)$(RetryLoop.pos=1)=IVRCallsPerIntv(CallType,Month,Interval,Days);


loop (RetryLoop$(RetryLoop.pos>1),
         AbandonCalls(RetryLoop,CallType,month,Interval,Days)$(AbandonRetry(RetryLoop-1,CallType,month,Interval,Days)>AbandRetryPrecision)
                                 =AbandonRetry(RetryLoop-1,CallType,Month,Interval,Days)*CallAbandonPct;
         AbandonRetry(RetryLoop,CallType,month,Interval,Days)$(AbandonRetry(RetryLoop-1,CallType,month,Interval,Days)>AbandRetryPrecision)
                                 =AbandonCalls(RetryLoop,CallType,month,Interval,Days)*CallRetryPct;
         Calls2PBX(RetryLoop,CallType,month,Interval,Days)$(AbandonRetry(RetryLoop,CallType,month,Interval,Days)>0)
                                 = Calls2PBX(RetryLoop-1,CallType,month,Interval,Days)- AbandonCalls(RetryLoop-1,CallType,month,Interval,Days)
                                 + AbandonRetry(RetryLoop-1,CallType,month,Interval,Days);
);
display IVRCallsPerIntv,MaxIVRCalls,AbandonCalls,AbandonRetry,Calls2PBX;

* This section selects the lowest value from the recursive to identify the real net traffic that will be presented to the PBX from
* the calculated RetryLoop section above.
parameter NetCalls2PBX(CallType,month,Interval,Days);

NetCalls2PBX(CallType,Month,Interval,Days)=sum(RetryLoop$(RetryLoop.pos=1),Calls2PBX(RetryLoop,CallType,Month,Interval,Days));
loop ((RetryLoop,CallType,Month,Interval,Days)$(RetryLoop.pos>1 and Calls2PBX(RetryLoop,CallType,Month,Interval,Days)>0),
         if(Calls2PBX(RetryLoop,CallType,Month,Interval,Days)<Calls2PBX(RetryLoop-1,CallType,Month,Interval,Days),
                 NetCalls2PBX(CallType,Month,Interval,Days)=Calls2PBX(RetryLoop,CallType,Month,Interval,Days);
         );
);

display IVRCallsPerIntv,MaxIVRCalls,AbandonCalls,AbandonRetry,Calls2PBX,NetCalls2PBX;
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
```

Figure 2: Recursive Loop

The next stage was to find the Traffic Offered by interval by call type. In order to find the offered traffic, we had to multiply the Net Call Arrival Rate by the average handling time. This then produced the offered traffic by call type. GAMS calculated the total calls presented to the PBX (for informational purposes) and also calculated the Offered Traffic value to be used in the ErlangB calculation on a one percent service level. The output for the Maximum Traffic Offered per month and the overall Maximum Traffic Offered is shown below. The MaxMaxTraffic is used as the *E* in the Erlang B equation. In order to get this number, the Net Call Arrival Rate had to be computed by changing the Net Calls to PBX to a ratio of calls per second. The Net Call Arrival Rate multiplied by the Average Handling Time was then used to find the Traffic to Interval. The Traffic to Interval was used to ultimately find the Maximum Offered Traffic, *E*.

4

- Ultimately calculate the Maximum Traffic (to be used as E)

```
----    3287 PARAMETER MaxTrafficOfferedMth

1  16.62088,    2  16.85043,    3  20.71642,    4  16.22732,    5  14.36173
6  16.49490,    7  14.37192,    8  16.10304,    9  21.00981,    10 19.18021
11 18.45184,    12 21.72630


----    3287 PARAMETER MaxMaxTraffic        =     21.72630
```

Next, we had GAMS calculate the final number of trunks to be used in the model. Again, we implemented a loop in order to find the number of trunks. This part of the code is the Erlang B calculation implemented into GAMS. The loop stopped at the probability cut off.

5

- Erlang B calculation to find Minimum Number of Trunks

```
----    3338 PARAMETER NumofTrunks          =       32.00000
```

The final portion of our GAMS model was the actual optimization model. Ultimately, this model needed to minimize the cost based on the number of trunks needed. Three types of trunks are in the consideration set. These trunk types are DS0, DS1, and DS3. Each trunk type has a different cost and a different capacity of channels. Therefore, our model had to have two parameters to show their differences.

The first parameter, TrunkType2Channels(TrunkType), specifies how many channels each trunk type can hold. DS0's have 1 channel; DS1's have 24 channels; DS3's have 672 channels. The second parameter, TrunkType2Cost(TrunkType), specifies the cost of each trunk type. DS0's cost $30.00 per month; DS1's cost $495.00; DS3's cost $4,000.00 per month.

Four variables are needed in this model. The four variables are TrunkSelected, TrunkTypeSelected, Z, and TrunkChannels. Each of these variables (except Z, which is just a free variable in place of the objective function) has dimensions NtwkTrunks and Month. The first equation is the objective equation - the Cost equation, which multiplies the cost of each trunk type by the trunk type selected. The next equation, MeetServiceObj, provided all the eligible trunk values. This was specified by setting the probability multiplied by the trunk selected to be less than the Erlang B service level of one percent. The third equation, TrkSelected, specified that only one trunk level could be picked per month. This was specified by making the binary variable, TrunkSelected, equal to one. The fourth equation provided the total number of trunk channel needed. Finally, the fifth equation, Combination, linked the capacity of various trunks to the objective equation by determining which type of trunk will meet the total channel need by month. This equation forced the objective to pick a combination of trunk types that sum to more than the minimum service level.

A snapshot of the model is shown in Figure 3 below:

6

- Optimization

```
SETS
      TrunkType / DS0, DS1, DS3 / ;
PARAMETERS
      TrunkType2Channels(TrunkType)  how many trunks are included in 1 purchase
        /    DS0        1
             DS1        24
             DS3        672    /
      TrunkType2Cost(TrunkType)  cost of each trunk type
        /    DS0        30
             DS1        495
             DS3        4000  / ;

Binary Variable         TrunkSelected(NtwkTrunks,Month)
Integer Variable        TrunkTypeSelected(Month,TrunkType)
Free Variable           Z        total trunk costs
Variable                TrunkChannels(NtwkTrunks,Month)

EQUATIONS
Cost              define objective function
MeetServiceObj(NtwkTrunks,Month)
TrkSelected(Month)
TrkChannels(NtwkTrunks,Month)
Combination(NtwkTrunks,Month)
;

Cost ..                              Z=E=SUM((Month,TrunkType),TrunkType2Cost(TrunkType)*TrunkTypeSelected(Month,Tru

* This gives all the eligible trunk values
MeetServiceObj(NtwkTrunks,Month) ..    TrunkSelected(NtwkTrunks,Month)*Probability(NtwkTrunks,Month)=L=ErlangBServLeve

* Specify can only pick one trunk level per month
TrkSelected(Month) ..                SUM(NtwkTrunks,TrunkSelected(NtwkTrunks,Month))=E=1;

* This gives the total trunk channels needed
TrkChannels(NtwkTrunks,Month)..      TrunkChannels(NtwkTrunks,Month)=E=TrunkSelected(NtwkTrunks,Month)*NtwkTrunks.po

* Determines what type of trunktype will meet the total channel need by month
* Link the capacity of various trunk types to the objective equation
* Forces the objective to pick a comination of trunktypes that sum to more than the minimum service level
Combination(NtwkTrunks,Month) ..
         TrunkChannels(NtwkTrunks,Month)  =L= SUM((TrunkType),TrunkType2Channels(TrunkType)*TrunkTypeSelected(Month,Tru

Model  ErlangBModel/ALL/;
ErlangBModel.optcr = .001;
Solve ErlangBModel Using MIP minimizing Z;

Display Z.l,TrunkSelected.l,TrunkChannels.l,TrunkTypeSelected.l;
```

Figure 3: Snapshot of Model Code

ANALYSIS AND MANAGERIAL INTERPRETATION

For the data set we used we obtained an optimal low cost of $7,140. The trunk types are as follows:

MONTH	DS0	DS1	TOTAL TRUNKS PER MONTH
January	2	1	26
February	2	1	26
March	7	1	31
April	2	1	26
May	-	1	23
June	2	1	26
July	-	1	23
August	1	1	25
September	7	1	31
October	5	1	29
November	4	1	28
December	8	1	32
TOTAL ANNUAL TRUNK COST			**$7,140.00**

The flexibility to change the amount of trunks is monthly. This means that if it is predicted that a call center will use less trunks in May, it could cut down on the number of trunks it uses for that month. Alternatively, for December you would utilize more trunks.

The number of trunks used per month will vary depending on the size of the call center, the amount of different call types it handles, and their average handling time. Our model can be used to help ORM's clients cut or add trunks on a month to month basis based on their history of call intensity.

Specifically, for the data set we used to build our model, the client likely paid for 32 trunks each month. If DS0s were used this would cost $11,520. Using DS1s the cost would go up higher. The biggest opportunity in using our model is the ability to change the number of trunks used monthly and then finding the optimal combination of trunk types, DS0 and DS1, to minimize cost.

CONCLUSIONS AND CRITIQUE

ORM Technologies operates a system utilizing uses web based inputs that allow the customer to input data and generate optimized resource requirements. It runs an Erlang C profile for every agent level in the pre-model and selects the minimum headcount required by 30 minute intervals by day, week, month and year.

Our team had the task of using Erlang B probability to dynamically estimate and model the most cost effective number of trunks, or phone channels, needed to service a live call center by trunk type. Some of the constraints associated with the model include:

1. Its language; it must be written in GAMS, General Algebraic Modeling System. Our team has no prior experience with GAMS.
2. Erlang B must be used as it can be integrated into the client's current model.
3. Time; the project has a 4 month time constraint

Our GAMS model does have some limits; it does not account for virtual connections like Session Initiation Protocol (SIP) trunks. Use of virtual connections allow the client to obtain a more dynamic cost model as capacity can be increased or decreased with a change in software. Capacity demands do not have to be forecasted in advance as they do with Primary Rate Interface (PRI) trunks.

We found the lowest annual trunk cost to be $7,140, with cyclic peak trunk usage at the close of every calendar quarter. Based on this, we recommend the client review their business needs quarterly.

WEEK ONE PROGRESS REPORT

Tue, Feb 4 – Mon, Feb 10

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we met with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

We discussed the background of their company and the processes in which they need assistance. ORM Technologies currently has a client with a call center that is looking to make their call center more efficient. Currently, ORM Technologies has a functioning system that uses web based inputs allowing the customer to input data and generate optimized resource requirements. The system runs an Erlang C profile for every agent level in the pre-model. From there, the optimization selects the minimum headcount required by 30 minute intervals by day, week, month and year. Some models are 12 months, but most are 5 years.

They typically have 1-5 call types and 1-10 agent types that have different rules and restrictions. The current blending approach recalculates the volumes by call type on available capacity. Secondly, it tests several levels of transfer of the call volume, which creates 5 or 6 times more choices to select from for blending. Upon completion of this model, the results can then be passed to a Shift Scheduling Manager that determines the optimal shift balance to satisfy the blended or unblended requirements.

The current system is a web-based client interface that runs at Amazon Web Services where the client data is captured and results presented. The Amazon server formats the new inputs into a package. This package file is then passed to ORM's server in Dallas where the optimization models reside (GAMS-based model) and the run is executed. From there, ORM passes the results back to the Amazon server where a number of presentation calculations are performed and then rendered to the client.

Two areas of this system need enhancement, which our team will provide:

1. The first requirement we will complete is to add Erlang B provisioning calculations and integrate these calculations into their current framework for dynamic modeling. Erlang B is a call-center modeling probability formula that can be used to calculate any one of the following three factors if you know or predict the other two:
    (i) Busy Hour Traffic (BHT): the number of hours of call traffic during the busiest hour of operation

(ii) Blocking: the percentage of calls that are blocked because not enough lines are available

(iii) Lines: the number of lines in a trunk group.

2. The second requirement for our team will be to build in the capability of blending. Currently the system has only a 1:1 blend type. ORM wants to know how to best implement '1-to-many' blend type or 'many-to-many' blend type. They would like for this blending approach to be implemented into GAMS as well.

We plan to meet with ORM's client next week in order to get a better understanding of the call center process. In the mean-time, ORM Technologies will be sending us real data so we can begin to familiarize ourselves with their data. While we are waiting to meet with them, our team will research algorithms and equations related to call-center optimization. We will also familiarize ourselves with GAMS programming language and the probability equations for Erlang B and Erlang C.

Team ORM Week 1 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Develop preliminary model outline**					
03.00.00	**Determine data requirements**					
04.00.00	**Construct prototype model**					
05.00.00	**Collect sample data**					

WEEK TWO PROGRESS REPORT

Tues, Feb 11 – Mon, Feb 17

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we met with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

Unfortunately, we were not able to meet with our client due to scheduling conflicts. We are still waiting for a data set to work with. While we are waiting to meet with them, our team will research algorithms and equations related to call-center optimization. We will also familiarize ourselves with GAMS programming language and the probability equations for Erlang B and Erlang C.

Team ORM Week 2 Progress Report Work Breakdown Structure	
01.00.00	**Acquire Client Requirements**
01.01.00	Meet with project advisor
01.02.00	Obtain project approval
02.00.00	**Develop preliminary model outline**
03.00.00	**Determine data requirements**
04.00.00	**Construct prototype model**
05.00.00	**Collect sample data**

WEEK THREE PROGRESS REPORT

Tues, Feb 18 – Mon, Feb 24

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

Unfortunately, we were not able to meet with our client due to their workload demands. We are still waiting for a data set to work with. While we are waiting to meet with them, our team will research algorithms and equations related to call-center optimization. We will also familiarize ourselves with GAMS programming language and the probability equations for Erlang B and Erlang C.

Team ORM Week 3 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Develop preliminary model outline**					
03.00.00	**Determine data requirements**					
04.00.00	**Construct prototype model**					
05.00.00	**Collect sample data**					

WEEK FOUR PROGRESS REPORT

Tues Feb 25- Mon, Mar 3

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

Unfortunately, we were not able to meet with our client due to their workload demands. We are still waiting for a data set to work with. While we are waiting to meet with them, our team will research algorithms and equations related to call-center optimization. We will also familiarize ourselves with GAMS programming language and the probability equations for Erlang B and Erlang C.

To give us a visual aid of the routing path of inbound calls, John sent us a diagram of the inbound call flow at their client's call center. He also provided us with an Erlang C calculator. Erlang C is used to determine the minimum number of required agents to meet the client's service level objective—80% of inbound calls answered within 30 seconds. It was sent with the intent to give us a guide as to how to formulate an Erlang B calculator.

A data set is still pending.

Team ORM Week 4 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Develop preliminary model outline**					
03.00.00	**Determine data requirements**					
04.00.00	**Construct prototype model**					
05.00.00	**Collect sample data**					

WEEKS FIVE AND SIX PROGRESS REPORT

Tues, Mar 4 – Mon, Mar 17

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

Unfortunately, we were not able to meet with our client due to the combination of their workload demands and Spring Break. While we are waiting to meet with them and obtain a data set, our team will research algorithms and equations related to call-center optimization. We will also familiarize ourselves with GAMS programming language and the probability equations for Erlang B and Erlang C.

Team ORM Weeks 5 and 6 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Develop preliminary model outline**					
03.00.00	**Determine data requirements**					
04.00.00	**Construct prototype model**					
05.00.00	**Collect sample data**					

WEEK SEVEN PROGRESS REPORT

Tues, Mar 18 – Mon, Mar 24

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

With only half a semester left, and no data set to work with, our group become concerned. We made a mid-term oral presentation to Dr. Barr and his Senior Design class on Tuesday, March 18 explaining:

- The background and business of ORM-Technologies
- The current state of the client, their constraints, and requirements, and
- Our group's current state in the project.

At that time, we were still awaiting a data set.

With that in mind, Ashley emailed John Ryan on Wednesday, March 18, explaining the urgency for data, reemphasized our group's enthusiasm and interest in the project, and requested a data set be received by no later than March 28. He understood. From there, two more emails were sent between our group and John Ryan. One from John explained that ORM was working with the client to obtain a data set for us. Neimy then emailed John requesting a meeting at the ORM office on:

- Monday, March 24th 12:30-1:30PM
- Tuesday, March 25th 12:30-3:30PM, 5:00-6:30PM

Unfortunately, that meeting did not occur due to prior obligations and scheduling conflicts. However, we did receive a data set from John on Monday, March 24.

Team ORM Week 7 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Develop preliminary model outline**					
03.00.00	**Determine data requirements**					
04.00.00	**Collect sample data**					
05.00.00	**Construct prototype model**					

WEEK EIGHT PROGRESS REPORT

Tues, Mar 25 – Mon, Mar 31

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

As stated in the Week 7 Progress Report, we were not able to meet with John at his office; however, we did reschedule to meet with him on April 1 at his office. Until then, we reviewed and tried to interpret the data set received. It had no row or column labels, so it was unclear as to exactly what we were reading.

To best maximize our hour with John on April 1, the three of us decided to meet on Sunday, March 30 at 8:30 AM to compile questions for John on the first. After the first Sunday meeting, our group began meeting every Sunday from 8:30AM to approximately 11AM to make progress on our project.

We decided our questions would be:

1) What is the data set measuring? Example, calls/day, calls/week, calls/month, etc?
2) How are we to structure our Erlang B model?
3) What are the inputs?
4) How are we to obtain the telecommunications pricing for the pricing model?

Team ORM Week 8 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Develop preliminary model outline**					
03.00.00	**Determine data requirements**					
04.00.00	**Collect sample data**					
05.00.00	**Construct prototype model**					

WEEK NINE PROGRESS REPORT

Tues, Apr 1 – Mon, Apr 7

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

Our meeting with John went great on April 1. He answered our questions as follows:

1. *What is the data set measuring? Example, calls/day, calls/week, calls/month, etc?*

   The data is measuring inbound calls for:
   - Every thirty minute interval from 7AM to 6:30PM
   - Over 60 months
   - Monday through Friday

2. *How are we to structure our Erlang B model?*

   Determine the peak call volume in the given data set. Use that value to construct the Erlang B model. Logically, if the model will satisfy the peak call volume over 60 months, it will satisfy the lesser call volumes over all 30 minute intervals.

3. *How are we to obtain the telecommunications pricing for the pricing model?*

   Trunk, or channel, types are DS0, DS1, DS3, and Sonet (fiber). He asked that we look up the average cost for each type, but was very clear in expressing that our cost model be built using the landline, DS0, pricing.

   NOTE: Later in the week, on April 6, John sent us a GAMS model and price points for each trunk type:

   DS0 - $30/month
   DS1 - $300/month
   DS3 - $5,000/month

4. *Could you explain the inbound call path diagram you sent?*

   The data set is representative of the call that reach the Wide Area Network (WAN) before they are routed to the various call center locations.

Before the calls reach the WAN:

1) The initial inbound call volume experiences are 5% abandon rate with a 1% retry. This means five percent of callers hang up or get disconnected before getting to the Private Branch Exchange (PBX) or Interactive Voice Response (IVR); one percent of this five calls back.

2) The remaining calls, or 95% of them route to the PBX.
    a. Out of the PBX, 95% of the 95% are routed to the IVR; this is the automated prompt service customers hear.
    b. Five percent of the 95% bypass the IVR and automatically route to a router for the WAN.
    c. Seven percent of the 95% of calls that come out of the PBX are resolved by the IVR and direct to a router.
        i. The router referred to in part (b) and part (c) directs these calls to the Wide Area Network for distribution across call center locations and home agents.

Overall, we left the meeting with a high level overview as the approach we are to take with the project.

1) Backward engineer the call volume from the provided data set to determine the gross call volume
2) Test inputs in online Erlang B calculator and in Excel.
3) Build front end of model.
    a. Pre-model Erlang in Excel
    b. Optimize cost per trunk by trunk type.

Team ORM Week 9 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Determine data requirements**					
03.00.00	**Collect sample data**					
04.00.00	**Develop preliminary model outline**					
04.01.00	Backward engineer data set to determine gross call volume					
04.02.00	Test Erlang B calculator online and in Excel					
04.03.00	Build front end of model					
04.03.01		Pre-model Erlang B in Excel				
04.03.02		Optimize cost per trunk by trunk type				
05.00.00	**Construct prototype model**					

WEEK TEN PROGRESS REPORT

Tues, Apr 8 – Mon, Apr 14

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

We received an Excel Erlang B calculator. During our group's weekly Sunday meeting, we worked with the Excel Erlang B calculator as well as an online Erlang B calculator. The online calculator measured trunks in hours, whereas the Excel calculator measured 30 minutes. After converting our time units to the same measurement, we were able to determine the optimal number of trunks needed to service the data received.

However, the greatest uncertainty with the project is writing the model in GAMS as we are all novices. Fortunately, with the help of a classmate, we were able to take the Excel model and write in Python.

Team ORM Week 10 Progress Report Work Breakdown Structure						
01.00.00	**Acquire Client Requirements**					
01.01.00	Meet with project advisor					
01.02.00	Obtain project approval					
02.00.00	**Determine data requirements**					
03.00.00	**Collect sample data**					
04.00.00	**Develop preliminary model outline**					
04.01.00	Backward engineer data set to determine gross call volume					
04.02.00	Test Erlang B calculator online and in Excel					
04.03.00	Build front end of model					
04.03.01		Pre-model Erlang B in Excel				
04.03.02		Optimize cost per trunk by trunk type				
05.00.00	**Construct prototype model**					
05.01.00	Erlang B Excel calculator					
05.02.00	Online Erlang B calculator					
05.03.00	Erlang B model written in Python					

WEEK ELEVEN PROGRESS REPORT

Tues, Apr 15 – Mon, Apr 21

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

On April 15, we met with our project advisor, Dr. Barr to discuss our April 1 meeting with John and our Erlang B model written in Python. Later that evening, on April 15th, Dr. Barr sent our group an Erlang B GAMS model.

We also analyzed our data set with Dr. Barr in more detail. Though John clarified that the data was a measurement of 60 months for every 30 minute interval from 7AM to 6:30PM, Monday through Friday, I was not clear if the data was a sum, average, or maximum of each day, Monday through Friday, or each week (i.e. the sum, average, or peak call volume of all Mondays per month or each Monday per month).

Team ORM Week 11 Progress Report Work Breakdown Structure		
01.00.00	**Acquire Client Requirements**	
01.01.00	Meet with project advisor	
01.02.00	Obtain project approval	
02.00.00	**Determine data requirements**	
03.00.00	**Collect sample data**	
03.01.00	Clarify data set more thoroughly	
04.00.00	**Develop preliminary model outline**	
04.01.00	Backward engineer data set to determine gross call volume	
04.02.00	Test Erlang B calculator online and in Excel	
04.03.00	Build front end of model	
04.03.01		Pre-model Erlang B in Excel
04.03.02		Optimize cost per trunk by trunk type
05.00.00	**Construct prototype model**	
05.01.00	Erlang B Excel calculator	
05.02.00	Online Erlang B calculator	
05.03.00	Erlang B model written in Python	

WEEK TWELVE PROGRESS REPORT

Tues, Apr 22 – Mon, Apr 28

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

On April 22, we met with our project advisor, Dr. Barr to discuss the prior week's progress and challenges. During this week, communication with John has increased. He provided an Erlang B GAMS calculator model to follow.

Team ORM Week 12 Progress Report Work Breakdown Structure		
01.00.00	**Acquire Client Requirements**	
01.01.00	Meet with project advisor	
01.02.00	Obtain project approval	
02.00.00	**Determine data requirements**	
03.00.00	**Collect sample data**	
03.01.00	Clarify data set more thoroughly	
04.00.00	**Develop preliminary model outline**	
04.01.00	Backward engineer data set to determine gross call volume	
04.02.00	Test Erlang B calculator online and in Excel	
04.03.00	Build front end of model	
04.03.01		Pre-model Erlang B in Excel
04.03.02		Optimize cost per trunk by trunk type
05.00.00	**Construct prototype model**	
05.01.00	Erlang B Excel calculator	
05.02.00	Online Erlang B calculator	
05.03.00	Erlang B model written in Python	

WEEK THIRTEEN PROGRESS REPORT

Tues, Apr 29 – Mon, May 8

Team ORM is composed of Lexi Farrar, Ashley Hall, and Neimy Sarmiento. On Tuesday, February 4, 2014 we had our initial meeting with our client, ORM Technologies. Dr. Barr, the ORM President, John Ryan, and the ORM Vice President, Ryan Cooper, all attended the meeting.

The team meet with John at his office on April 29. To simplify our approach, he provided a MultiCall file with multiple call types spanning only 12 months of data. He also provided us with a SingleCall file; it included 120 months of data, but on only one call type. Both files included all the necessary parameters like Average Handle Time and number of months. Finally, in preparation for the final presentation, John sent us a PowerPoint with a few strong visual aids and ORM company background information.

At our weekly team meeting on Sunday, May 4, we assigned duties for the final written report as follows:

1) One-page Management Summary: Ashley
2) Background and Description of the Problem Situation: All
3) Analysis of the Situation: Neimy
4) Technical Description of the Model: Lexi
5) Analysis and Managerial Interpretation: Neimy
6) Conclusions and Critique: Ashley

All parts are to be completed by the morning of Wednesday, May 7, 2014.
The group will make its 15 minute oral presentation of our findings on Thursday, May 8, 2014 between 11:30AM and 2:30PM. In attendance will be the Spring 2014 EMIS 4395: Senior Design Project Advisors, Dr. Dick Barr and Dr. Tom Siems, fellow Management Science Senior Design students and underclassmen, and members of the EMIS Department Faculty.

Team ORM Week 13 Progress Report Work Breakdown Structure	
01.00.00	**Acquire Client Requirements**
01.01.00	Meet with project advisor
01.02.00	Obtain project approval
02.00.00	**Determine data requirements**
02.01.00	Average Handle Time
02.01.00	Average call volume per 30 minute interval
03.00.00	**Collect sample data**
03.01.00	Clarify data set more thoroughly
03.02.00	Use MultiCall file
03.03.00	Use SingleCall file
04.00.00	**Develop preliminary model outline**
04.01.00	Backward engineer data set to determine gross
	call volume
04.02.00	Test Erlang B calculator online and in Excel
04.03.00	Build front end of model
04.03.01	Pre-model Erlang B in Excel
04.03.02	Optimize cost per trunk by trunk type
05.00.00	**Construct prototype model**
05.01.00	Erlang B Excel calculator
05.02.00	Online Erlang B calculator
05.03.00	Erlang B model written in Python
05.04.00	Erlang B written in GAMS to include cost analysis for
	dynamic pricing