

## ABSTRACT

### DESIGN AND IMPLEMENTATION OF AN EMPLOYEE JOB SCHEDULING AND PROGRESSION TRACKING SYSTEM

by Michael C. Stahr

This thesis is a project-oriented one that features the design and development of a software system used for employee hourly progression tracking and job scheduling in an industrial environment. This software was developed at a professional level as verified by its adoption by one of the largest boxboard companies in the industry for its day-to-day operations. Some of the main features of this system are:

- ✍ Specialized ActiveX controls
- ✍ Object oriented design utilizing classes in Visual Basic 6.0 and C++
- ✍ Stored procedures embedded in Microsoft SQL Server 7.0
- ✍ TCP/IP technology

This system consists of two integrated subsystems, Tracker and Crew Master. The first, Tracker, was designed to replace a manual system used by the accounting department for tracking employee hourly progression. The second, Crew Master, replaces a manual system for scheduling hourly employees for job positions as well as vacation scheduling.

DESIGN AND IMPLEMENTATION OF AN EMPLOYEE JOB SCHEDULING  
AND PROGRESSION TRACKING SYSTEM

A Thesis

Submitted to the

Faculty of Miami University

In partial fulfillment of

The requirements for the degree of

Master of Systems Analysis

Department of Computer Science and Systems Analysis

by

Michael C. Stahr

Miami University

Oxford, Ohio

2002

Advisor\_\_\_\_\_

Dr. James Kiper

Reader\_\_\_\_\_

Dr. Tom Schaber

Reader\_\_\_\_\_

Dr. Alton Sanders

## TABLE OF CONTENTS

1.	Introduction.....	1
2.	The Project Overview.....	3
3.	The Proposal.....	6
4.	Important Features.....	10
5.	Requirements.....	16
6.	The Database.....	17
7.	Conclusion.....	18
8.	Appendix A.....	19
9.	Appendix B.....	24
10.	Appendix C .....	29
11.	Appendix D.....	45
12.	Appendix E.....	47
13.	Appendix F.....	57

## **I. Introduction**

Industrial America has long been run by manual administrative systems fashioned to fulfill the needs of day to day business activity. Over the past two decades we see a paradigm shift to desktop database applications created to replace these archaic manual systems. We have moved from paper and pencil to more sophisticated computer programs designed to simplify workflow and increase productivity.

As surprising as it may seem, there are still many legacy systems driving day-to-day business in corporate America today. These legacy systems offer a sense of security but typically fall short in the areas of productivity and reasonable flexibility. The idea “if it’s not broken, don’t fix it” has prevented some organizations from utilizing their full potential. These companies continue to rely on the “paper and pencil” method because, however slow it may be, the system works. Resistance to change becomes one of the most difficult hurdles to get over when implementing computer software solutions into a company with such an archaic system.

As with any organization, there are many problems that can hinder the efficiency of its internal systems. Above, I mention two ways of conducting business: paper and pencil, and computer designed software. This thesis project focused on the design and implementation of a computer software system for a

manufacturing facility which was designed and created to replace long standing manual administrative processes.

## II. THE PROJECT OVERVIEW

A large boxboard manufacturing company requested a system to be created that would replace many manual systems that have been an integral part of its payroll and labor tracking processes for the past three decades. My first introduction to their legacy time keeping system came in the form of a binder book. The manufacturing facility for which I initially developed these systems literally referred to this as “The Black Book.” The purpose of this book was to track hours worked for every employee per machine positions on a week-to-week basis. The Black Book was a manual system created to store weekly information per machine position in order to evaluate the employee’s progression (or level of expertise) per union contract. An employee would be paid a rate based on their level of progression. Upon reaching a new level of progression (also called a *Labor Grade*) the employee would be given an increase in hourly pay for the particular machine position to which they had progressed. So, for instance, once an employee worked a total of 1040 hours on *Labor Grade 10-33* (also called an *assistant pressman* or *second pressman*) they would move up to the next labor grade which would be *21-36* (also called a *first pressman*.) The employee would then be paid the new rate for that machine position based on the union contract book.

Although this may seem, at first glance, to be a straight forward execution of tasks, there are many “loop-holes” to the process. For instance, employees, at

times, are granted “special” exceptions to the contract rules that offset their requirements for progression and pay. In these situations only individuals (for instance a payroll clerk) with this knowledge can anticipate how to pay these employees. All time and progression tracking sheets (Appendix A) were manually input into The Black Book for every employee that worked during the week. The complexity of a new payroll clerk being hired and understanding this system was overwhelming and, therefore, was not an option for more than twenty years! There were a total of two payroll individuals who were in charge of progression tracking for more than twenty years and one of the two was only there to help during overload times.

In addition to the complexity of tracking labor grades, errors in the system were often accepted and anticipated. Every few months there would be a scheduled weekend of overtime dedicated to balancing the employee’s actual worked hours. With this manual system it was expected that there would be at least two weeks of undocumented time tickets to be entered into The Black Book at any given time. At this point any individuals paid incorrectly would be given a back-pay check. The fact that this was a common practice at the facility caused many grievances and union disputes due to incorrect employee paying. As the plant grew in the number of hourly employees, the problems for the accounting department increased as well.

Another area of concern for the company was employee work and vacation scheduling. Every floor supervisor was, and is, required to schedule every

employee in their department. This, too, is based on union contract and special arrangements made to various employees.

Scheduling employees for machine positions was one of the most diverse operations at the plant level. Each supervisor had developed their own method for scheduling employees (Appendix B) which was the basis for everything from scheduling to vacation seniority to layoffs. Issues arose constantly with this chaotic system. Employees constantly transfer between departments as well as trading their scheduled position with one other. This does not even take into account the chaos that occurs when a supervisor is out for the day. In these cases a covering supervisor takes over the department but has little knowledge of the actual schedule.

The scheduling process also has to work with the accounting department. Time tickets have to be reconciled with the scheduled position for every employee if they are to be paid correctly as well as progress on their scheduled machine position. As you can tell, these two systems coordinate with one another and, therefore, must work seamlessly. Floor supervisors must retain their current level of flexibility in scheduling employees and the accounting department must be able to track employee progression while keeping with a constantly changing union rule book.

I was brought in to computerize these processes. The goals of this project include a computer system to simplify scheduling employees and tracking labor



and would hopefully have the side effect of reducing or eliminating labor disputes and reliance on single individuals.

## **THE PROPOSAL**

In reviewing the current system I determined there were several areas within the company in which individuals need to manipulate data and for which it would be advantageous to develop multiple desktop systems. In fact, I proposed the development of two applications to fulfill the needs of the client. The first system, *Tracker*, would be designed to input hourly time tickets, track employee progression, and interact with various systems such as PeopleSoft® and Kronos®. The second application, *Crew Master*, would aid floor supervisors in scheduling employees for machine positions as well as vacation. The system accounts for all employees so that individuals would be unable to “fall between the cracks” and get paid for work they did not do or miss pay for work that they did accomplish.

Although the accounting department would gain the most from a computer system, I found other departments, such as personnel, and the human resource group could benefit from the proposal as well. The decision to take the route of two separate but integrated applications was also influenced by that fact that a large software system tends to be overwhelming for both a developer as well as the end user.

There were many advantages to developing multiple systems to fill the needs of the client. One, of course, was the simple fact that smaller applications are

easier to maintain through the development process. I found I was able to create these applications quickly by concentrating on one system at a time.

One of the problems in creating multiple systems is that it is harder to develop the entire “back-end”, or database, system structure at one time. Creating a solid database design is much easier when all elements are proposed at one time. I found that by creating a system for the accounting department (*Tracker*) in such a way that was somewhat independent of the employee scheduling system (*Crew Master*) caused database issues later in the design process of *Crew Master*. For instance, the common language for the accounting department in regards to absenteeism was quite a bit different than that of the department managers or even the human resource department.

Another problem with this method was the tendency to duplicate the lines of code between applications. Although much of this was cut down by creating system DLL's (dynamic linked libraries) or ActiveX controls there was still a tendency to simply “cut and paste” from one application to another in order to save time. The drawback to this was apparent when code changes needed to be made in copied segments of code.

At one point it was proposed to purchase an “off-the-shelf” system that could be modified to fix Smurfit's needs and to have less “in-house” development. The logic to this was based in the feeling that a purchased product would be more “stable.” The company tends to feel that a marketed system will always offer support and upgrades to the system they produced, whereas, a system developed

within the company has a risk of not being supported. The comical twist to this logic is that the IT staff strongly backed the idea that a system developed “in-house” poses as much or more stability than a product created by a third-party vendor. Just the fact that the facility owns the source code to any “in-house” system is justification enough. Management, however, has little understanding of source code.

As illogical as it may seem I was initially instructed to create the Tracker application using Microsoft’s Access VBA development environment. Visual Basic was a new “animal” to the company and it was felt that Access was a stable system that allowed the user to change how the system worked at will. This request was wrong on so many levels that I found it almost comical they would suggest such a method. My response to this was that they should find someone else to write the system. In the end I produced a document stating the disadvantages of using VBA in Access over using pure VB. The following is a list of some of the disadvantages of using VBA:

1. VBA is a watered down version of VB so you lose many features of a more robust development environment.
2. VBA does not support DLL or ActiveX creation.
3. VBA is much more difficult to code in than VB.
4. It is harder to connect to outside data sources using VBA.
5. Controls are bound to data base tables by default in VBA which creates much unnecessary overhead.

6. The VBA environment is not set up for “hard-core” programming. It was created so the average user could bind data controls to database tables and view data.

### **III. Important Features**

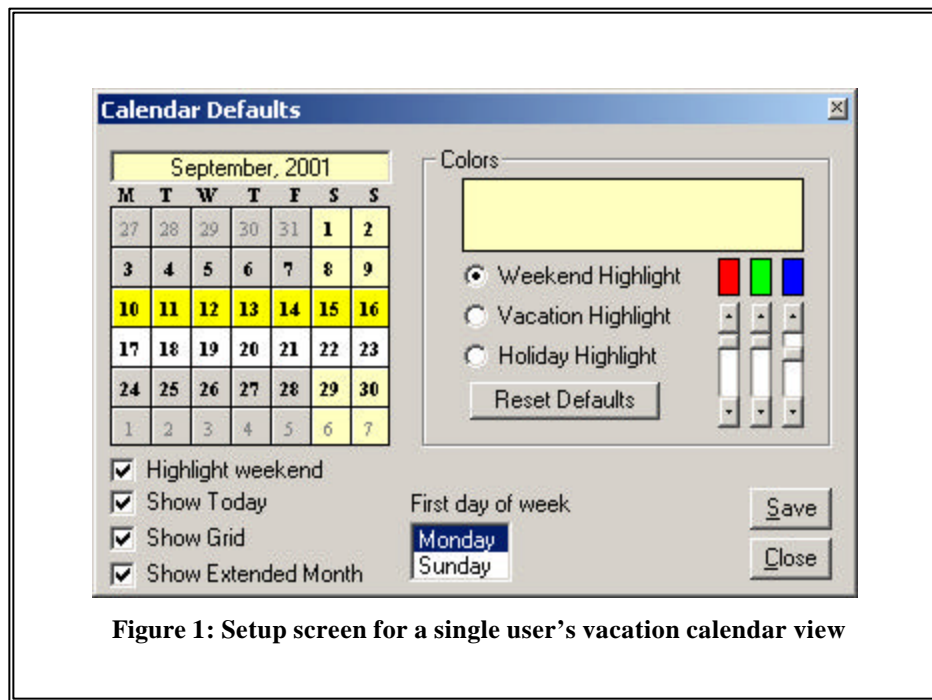
Both Tracker and Crew Master have been written with many elements seen in “off the shelf” products. The following is a comprised list of the most important features I feel the systems offer.

1. Specialized ActiveX controls.
2. Object oriented design utilizing classes in Visual Basic 6.0 and C++.
3. Stored procedures embedded in Microsoft SQL Server 7.0.
4. TCP/IP technology.

#### **A. Specialized ActiveX controls**

One of the more exciting things I did when creating Crew Master was to develop a custom ActiveX vacation control that encapsulated the company business logic. Figure 1 illustrates a screen containing the vacation control with customized features for each user of the system. The control itself displays unpaid vacation request by day or week in yellow. Company holidays are defaulted to white while weekends are painted in light yellow. Unpaid vacations turn green when the accounting department triggers the pay sequence of vacation through Tracker. Once a vacation has been paid a supervisor is unable to change the vacation through Crew Master. This control tracks the allocated vacation hours as well as the type of vacation employees are given and use. For example, an employee may be allocated 80 hours of vacation for the year and an additional 40 upon reaching their anniversary hire date. The control will only allow the employee to schedule up to 80 hours before their anniversary date and then the

additional 40 after their date. Some employees are given the option to schedule “pay only” vacation. Pay only is equivalent to a working vacation. The employee will receive a vacation pay check but will continue to work during the selected week. The ActiveX control monitors this allocation and prevents supervisors from making mistakes in the process.

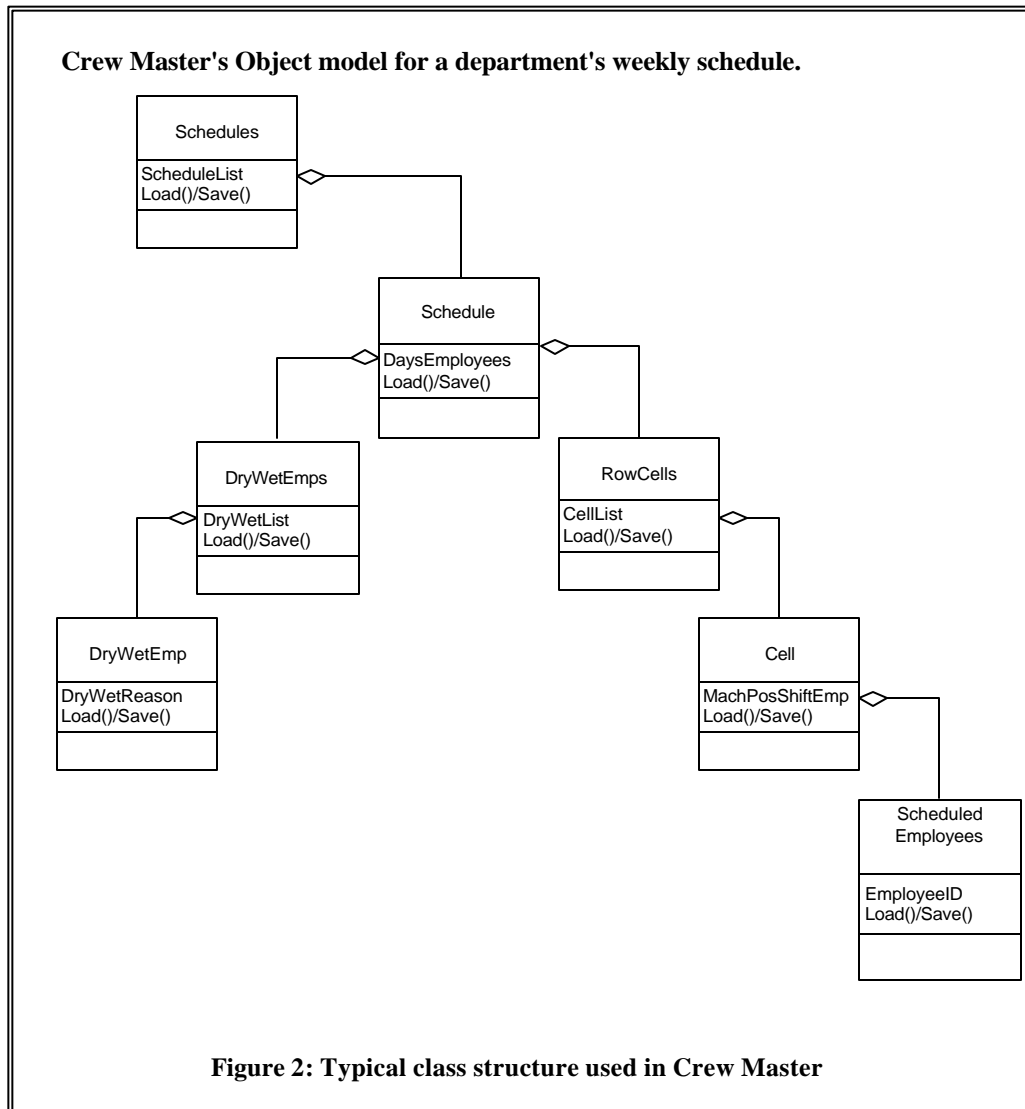


**Figure 1: Setup screen for a single user’s vacation calendar view**

## **B. Object oriented design utilizing classes in Visual Basic 6.0 and C++**

Object oriented design was an important feature to implement in both Crew Master and Tracker. The simple fact that both systems would eventually grow to be quite large, a solid class structure would be a necessity to drive the back end of the system.

I chose to create classes with accompanying collection classes that would almost mimic my database structure. Business logic was coded into these classes in order to expedite coding throughout the system. Figure 2 displays an example of the class structure of a departments schedule for a week.



### **C. Stored procedures embedded in Microsoft SQL Server 7.0**

One of best ideas used in *Crew Master* was to encapsulate all database queries into stored procedures. These stored procedures are server side based and are compiled routines residing in the SQL Server database. Creating stored procedures allows for changes to be made to the data retrieval stream without having to change the user interface. So, for instance, I created a desktop application to handle scheduling crews and vacation but it would be relatively simple to create a Web interface to the same system because all that would need to be done is to create screens that use the same data that the desktop system uses. The impact of this decision was not fully appreciated until deployment of the system to the multiple plants. I found that at several plants there were existing databases set up with current and historical data. Tables with machine information were already established and being used by other systems in various plants. Upon installing the Crew Master system I needed only to adjust a few stored procedures in order to utilize existing databases. For example: When installing Crew Master at our Carol Stream plant I was informed that they had a database (Production) that was currently being used by multiple in-plant systems. The Production database has tables such as: *Machines*, *Departments*, etc., and because Crew Master also has these tables (*tbl\_Machines*, *tbl\_Departments*, etc.) I only needed to change Crew Master's stored procedures to point to the Production database rather than having to change source code.



```

CREATE PROCEDURE spLoadScheduledCrew
@DMSKey int
AS
SELECT
    EmployeeSchedule.ESEmployee,
    EMP.LastName + ' ' + EMP.FirstName,
    EmployeeSchedule.ESPositionCode, PositionDescription
FROM
    EmployeeSchedule
INNER JOIN
    EMS.dbo.tbl_Employees EMP ON
    EmployeeSchedule.ESEmployee = EMP.PSID
INNER JOIN
    EMS.dbo.tbl_MachinePositions MP ON
    EmployeeSchedule.ESPositionCode = MP.MachinePosition_Key
WHERE
    (EmployeeSchedule.ESDMSKey = @DMSKey)
ORDER BY
    EMP.LastName + ' ' + EMP.FirstName

```

**Figure 3: SQL Server stored procedure from the Production database**

Along the same lines, I found that when it was time to link Carton Pro to Crew Master the only concern was to make sure all common stored procedures were updated. Carton Pro's involvement with Crew Master relies on retrieving

employee crewing schedules in order to apply job costing to designated crews as well as production logs. Therefore, Carton Pro's scaled-down version of a scheduling system was phased out when Crew Master was installed. Carton Pro's process of retrieving scheduling information was converted to stored procedures which now pointed to Crew Master's database. The advantage of using and changing a stored procedure is that the application doesn't care how the data is produced. A stored procedure's only purpose is to execute commands and/or return a database recordset with expected column names. Figure 3, above, is an example of a stored procedure from the Production database which loads a scheduled crew from the Crew Master database (EMS). Notice the reference to the EMS database via *EMS.dbo.tbl\_Employees* and *EMS.dbo.tbl\_MachinePositions*.

#### **D. TCP/IP technology**

One obstacle to overcome was the problem of synchronization with the database when multiple users would be using Crew Master at the same time. It would be very feasible that a supervisor could be reviewing their schedule while another supervisor transfers employees back and forth between departments. This posed a problem because a user may set up their schedule based on who they thought was in their department. If an employee was transferred out of a department while a supervisor was in the middle of setting up their schedule they wouldn't know they had just lost an employee. I chose to solve this issue by

creating a TCP/IP connection client and server with which Crew Master communicated. The Crew Master server would trigger any action that would require a broadcasted message and all client screens would be automatically updated. One issue, however, was that Crew Master could not rely on the server to be running all of the time; therefore, I created a synchronization procedure in Crew Master that would update screens each time a supervisor saved a change.

#### **IV. Requirements**

The following is a list of the system requirements that comprises the crewing / time collection and progression tracking system.

The System must fulfill the following requirements:

**1. Machine position scheduling per department and shift (Crew scheduling software – Crew Master)**

- ? This system should replace individual systems developed by various salaried employees in the plant.
- ? It is required that all employees are accounted for during a scheduled shift.
- ? Employees that are not scheduled for a machine position should be listed as “unscheduled” and given an unscheduled reason.

**2. Vacation scheduling for hourly employees**

? The system should track employee vacation and automatically move employees to the “unscheduled” list during vacation time.

Vacation time can fall in one of the following two categories:

“actual” or “pay only” vacation.

**3. Exception reporting for absentee/tardy employees integrated with current Kronos® system.**

? Kronos® is a time card swiping system used in place of a time ticket. The Kronos® output file should be reconciled with the proposed database records.

**4. Time and attendance tracking integrated with current corporate ER package, PeopleSoft®.**

**5. An extensive system to monitor and track labor grade progression for hourly employees (Tracker.)**

**6. Integration of the scheduling software with the current corporate Carton Pro system.**

**V. The Database**

Crew Master uses SQL Server 7.0 as its databank storage. Tracker uses Microsoft Access 2000. I proposed that all systems utilize the same database

(SQL Server) to maximize and streamline data flow. The client, however, required using a Microsoft Access database for the back-end of the Tracker system. This was due to a desire to produce interactive reports and “forms” in a system already mastered by the common company “user.”

## **VI. Conclusion**

In conclusion, Crew Master has been implemented into many plants in order to produce crewing teams. Tracker, although powerful, has only been implemented into a total of 5 plants. Many plants currently have their own system of tracking employee progression and are resistant to change. In time I anticipate more and more plants will utilize the advantages of Tracker.

Creating these systems has given me an incredible amount of knowledge and experience in both system development and corporate level manufacturing. During the development stages I was forced to research and perfect techniques I had previously avoided, such as TCP communication, and SQL Server stored procedures and views.

All-in-all, I truly enjoyed this project and have learned more than I can eloquently express.

Appendix A:  
Archaic Method for  
Time Collection and  
Progression Tracking

The following three figures denote a typical employee time tracking chart from

"The Black Book"

Figure 1:

8-19-91  
772

EMPLOYEE'S QUALIFIED RATES

New Hire 10/21/91 ✓

772

Clock Number NAME

JOB DESCRIPTION	GRADE		RATE RANGE			
	Labor	Job	Auto-Step	Fully Qualified	Step	Maximum
Floor Help	5	19	3 mts	1-5-92		
			6	3-22-92		
			9	7/30/92		
			12	11-15-92		
			15	8-15-93		
			18	10-31-93		
			21	7-6-94		
		24	9-11-94			

HOURS WORKED ON HIGHER RATED JOB

GRADE	Job	Lab.	Week Ending	Hours	Total To Date	Week Ending	Hours	Total To Date	Week Ending	Hours	Total To Date	Hours Required
5	19	Floor Help	10-27-91	46	46	4-5-92	25	71	12-11-91	22	93	
			11-3-91	48	94	4-14-92	31.5	124.5	1-7-93	32	127.5	
			1-10-92	46	140	4-26-92	32	152.5	2-2-93	56	178.5	
			4-17-92	46	186	5-9-92	46	232	2-14-93	24	256	
			6-24-92	46	232	5-21-92	62	294	6-22-93	48	342	
			8-31-92	23	255	7-31-92	32	326	7-19-93	40	386	
			10-29-92	56	311	8-2-92	76	387	7-25-93	32	419	
			12-15-92	72	457	6-23-92	48	435	8-2-93	36	471	
			12-29-92	92	451	7-14-92	38	473	8-15-93	68	541	
			1-12-93	32	483	8-27-92	56	529	8-22-93	76	605	
			1-5-93	43	526	8-16-92	68	594	8-22-93	68	662	
			1-12-93	48	574	8-24-92	80	674	9-5-93	48	722	
			1-19-93	48	622	9-13-92	32	704	9-12-93	56	780	
			1-26-93	76	698	9-26-92	50	748	9-19-93	48	796	
2-2-93	46	744	9-27-92	56	804	9-26-93	34	838				
2-9-93	46	790	10-4-92	48.5	852.5	10-24-93	46	898.5				
2-16-93	52	842	10-25-92	48	900	10-10-93	24	924				
2-23-93	52	894	11-1-92	48	948	10-24-93	46	994				
3-6-93	64	915	11-8-92	38	986	10-31-93	65	1051				
3-13-93	48	963	11-25-92	80	1063	11-2-93	56	1119				
3-20-93	43	1006	11-26-92	48	1111	11-16-93	48	1159				
3-27-93	46	1052	11-29-92	24	1135	11-21-93	64	1223				
4-3-93	43	1095	12-3-92	26	1161	11-28-93	26	1249				
5	31	Floor Help	11-29-91	8	8	7-25-93	16	88	12-19-93	12	100	
			12-5-91	16	24	8-5-93	12	100	12-26-93	16	116	
			12-12-91	8	32	8-12-93	46	104	1-2-94	46	150	
			12-19-91	8	40	8-19-93	40	144	1-9-94	32	182	
			12-26-91	8	48	10-2-93	24	168	1-16-94	48	230	
			1-2-92	8	56	10-9-93	8	176	1-23-94	32	262	
			1-9-92	8	64	10-16-93	8	184	1-30-94	40	302	

5203 mod

Date Eligible

10406

15609

Date Eligible

200 12 months

2600 15

Date Eligible

3120 (8)

3640 (21)

1040

6 months

Date Eligible

3695

Figure 2:

HOURS WORKED ON HIGHER RATED JOB											
Job Grade	Job No.	Week Ending	Hours	Total To Date	Week Ending	Hours	Total To Date	Week Ending	Hours	Total To Date	Hours Required
7 32	518 99	8-23-94	8	8	9-6-94	56	298	10-13-94	8	444.5	1040
		9-6-94	8	16	9-13-94	34	332	9-27-94	4	577.6	
		9-27-94	8	24	9-27-94	64	396	9-27-94	4	581.6	
		10-3-94	16	40	9-28-94	45	341.5	10-1-94	12	593.6	
		10-3-94	16	56	10-4-94	48	389.5	10-4-94	12	605.6	
		10-7-94	8	64	10-16-94	8	397.5	10-16-94	8	613.6	
		10-16-94	8	72	10-22-94	28	425.5	10-22-94	8	621.6	
		10-22-94	8	80	10-26-94	2	427.5	10-26-94	12	633.6	
		10-26-94	16	96	10-27-94	8	435.5	10-27-94	8	641.6	
		10-27-94	8	104	10-27-94	16	451.5	10-27-94	32	673.6	
5 19	Floor Help	10-27-94	8	112	1-2-95	16	467.5	1-1-95	16	483.6	7080 12
		1-2-95	8	120	1-2-95	16	483.5	1-1-95	32	515.6	
		1-2-95	8	128	1-2-95	16	500.5	1-1-95	16	531.6	
		1-2-95	8	136	1-2-95	16	516.5	1-1-95	16	547.6	
		1-2-95	8	144	1-2-95	16	532.5	1-1-95	16	553.6	
		1-2-95	8	152	1-2-95	16	548.5	1-1-95	16	564.6	
		1-2-95	8	160	1-2-95	16	564.5	1-1-95	16	580.6	
		1-2-95	8	168	1-2-95	16	580.5	1-1-95	16	596.6	
		1-2-95	8	176	1-2-95	16	596.5	1-1-95	16	612.6	
		1-2-95	8	184	1-2-95	16	612.5	1-1-95	16	628.6	
5 31	Help	1-2-95	8	192	1-2-95	16	628.5	1-1-95	16	644.6	1040
		1-2-95	8	200	1-2-95	16	644.5	1-1-95	16	660.6	
		1-2-95	8	208	1-2-95	16	660.5	1-1-95	16	676.6	
		1-2-95	8	216	1-2-95	16	676.5	1-1-95	16	692.6	
		1-2-95	8	224	1-2-95	16	692.5	1-1-95	16	708.6	
		1-2-95	8	232	1-2-95	16	708.5	1-1-95	16	724.6	
		1-2-95	8	240	1-2-95	16	724.5	1-1-95	16	740.6	
		1-2-95	8	248	1-2-95	16	740.5	1-1-95	16	756.6	
		1-2-95	8	256	1-2-95	16	756.5	1-1-95	16	772.6	
		1-2-95	8	264	1-2-95	16	772.5	1-1-95	16	788.6	
10 33	Cost Press man	1-2-95	8	272	1-2-95	16	788.5	1-1-95	16	804.6	1040
		1-2-95	8	280	1-2-95	16	804.5	1-1-95	16	820.6	
		1-2-95	8	288	1-2-95	16	820.5	1-1-95	16	836.6	
		1-2-95	8	296	1-2-95	16	836.5	1-1-95	16	852.6	
		1-2-95	8	304	1-2-95	16	852.5	1-1-95	16	868.6	
		1-2-95	8	312	1-2-95	16	868.5	1-1-95	16	884.6	
		1-2-95	8	320	1-2-95	16	884.5	1-1-95	16	900.6	
		1-2-95	8	328	1-2-95	16	900.5	1-1-95	16	916.6	
		1-2-95	8	336	1-2-95	16	916.5	1-1-95	16	932.6	
		1-2-95	8	344	1-2-95	16	932.5	1-1-95	16	948.6	



# EMPLOYEE'S QUALIFIED RATES

172  
Clock Number NAME

JOB DESCRIPTION	GRADE		RATE RANGE			
	Labor	Job	Auto-Step	Fully Qualified	Step	Maximum
App. Jucker	7	32	6 month	11-1395		
			12 month	11-2-97		
			13			
			24			
		40	30			

## HOURS WORKED ON HIGHER RATED JOB

GRADE	Job	Lab.	Week Ending	Hours	Total To Date	Week Ending	Hours	Total To Date	Week Ending	Hours	Total To Date	Hours Required
7	32	Jucker	11/1/96	32	1634.1	7-2-97	4	1719.1	11-2-97	16	2121.1	2080
			8-1-96	4	1638.1	7-22-97	20	1739.1	11-2-97	56	2187.1	2080
			8-25-96	4	1642.1	8-3-97	1	1940.1	11-22-97	72	2259.1	2080
			9-1-96	10	1658.1	8-10-97	4	1743.1	12-2-97	56	2315.1	2080
			9-5-96	10	1668.1	8-27-97	17	1761.1	12-14-97	64	2379.1	2080
			9-12-96	8	1676.1	8-31-97	22	1783.1	12-21-97	48	2427.1	2080
			10-3-96	2	1678.1	9-7-97	16	1799.1	1-2-98	48	2475.1	2080
			10-19-96	8	1682.1	9-20-97	27	1843.1	1-4-98	44	2519.1	2080
			11-1-96	5	1687.1	9-27-97	22	1851.1	1-11-98	2	2519.1	2080
			12-1-96	1	1688.1	9-28-97	48	1923.1	1-18-98	8	2531.1	2080
			1-22-97	5	1696.1	10-5-97	32	1955.1	1-25-98	39	2562.1	2080
			1-29-97	4	1700.1	10-12-97	48	2003.1	2-1-98	41	2604.1	2080
			2-6-97	4	1704.1	10-19-97	48	2043.1	2-8-98	46	2644.1	2080
			2-13-97	2	1706.1	10-26-97	16	2059.1	2-15-98	48	2684.1	2080
			2-19-97	4	1715.1	11-2-97	20	2115.1	3-5-98	8	2692.1	2080
			3-5-98	40	2732.1							
			3-29-98	40	2772.1							
			4-19-98	40	2812.1							
			4-12-98	24	2836.1							
			4-19-98	42	2858.1							
			5-3-98	5	2863.1							
			5-27-98	30	2909.1							
			6-24-98	7	2957.1							
			6-31-98	8	2965.1							
			7-2-98	2	2967.1							
			8-9-98	4	2971.1							
			8-11-98	2	2973.1							
			2-5-98	32	3005.1							
			3-1-98	56	3061.1							

Figure 4:

23

Appendix B  
Typical “Home grown”  
Department Schedules



Figure 1: Finishing Schedule

FINISHING DEPARTMENT SCHEDULE

REVISED    ##

WEEK OF:

12/13/99

M T W T F S S

1ST

2ND

M T W T F S S

3RD

M T W T F S S

L. PERSON:

F. Botos

OPERATORS:

D. Glenn

OT

W. Bailey

B. Peizel

OT

R. Gerner

W. Dunaway

OT

D. Bartrum

R. Swain

OT

T. Spicer

J. Butler(t)

OT

G. Pruitt (t)

A. Watkins(t)

OT

B. Flannery(t)

JA-PACKER

1 K. Robinson

J. Southall

W. Day

2 B. Hedger

S. Knott

S. James

3 D. Richie

B. Riley

C. Jackson

4 C. Wofford

X. Bustos

D. Rogers

5 G. Baker

M. Eby

C. Wilson

6 S. Frazier

G. Brown

M. Vance

7 J. Dunn

J. Howard

D. Smiley

8 R. Downing

J. Mullins

J. Emerson

9 D. Baughn

B. Kirkpatrick

L. Dietz

10 P. FELTNER

C. Milligan

K. Kurtz

11 J. Russell

G. Thompson

S. Menefee

12 M. Beckett

D. Hundley

T. Bernard(t)

13

14

PALLETIZER:

T. Hedger

L. King

G. Howse

CORR. MAT.

D. Bagley

S. Butler

S. Pearson

HAND.

L. Hunt (t)

PTO:

P. Green

D. Lawson

J. Tolson

R. Fordyce

W. Bowles

T. Morrison

R. Miller

SHIPPING:

P. Major

TRADES:

SICK:

G. Osborne

N. Manning

D. Wilson

VACATION:

T. Bowker

C. Edwards

M. Tackett

Figure 2: Flexo, Sheeter, Quality, and Maintenance Schedules

Week of 12/13/99																									
FLEXO DEPARTMENT																									
		1st	M	T	W	Th	F	Sa	Su	2nd	M	T	W	Th	F	Sa	Su	3rd	M	T	W	Th	F	Sa	Su
Operator	K. Philpot									B. Honchul								R. Combs							
Assistant	P. Higdon									D. Lawson								N. Boone							
Stackers	D. Brown									L. Zecher								C. Owens							
	C. Prichard									K. Belcher								J. Olinger							
Trades																									

		<u>Sick</u>		<u>Vacation</u>																				
		SHEETER																						
Week of	12/13/99	1st	T	W	Th	F	S	S	2nd	M	T	W	Th	F	S	S	3rd	M	T	W	Th	F	S	S
Operators		D. Benson							C. Carter								C. Brown							
		L. Haddix							S. Begley								J. Henning							

Sick  
Vacation  
Mill

		QUALITY																							
Week of	12/13/99		T	W	Th	F	S	S	2nd	M	T	W	Th	F	S	S	3rd	M	T	W	Th	F	S	S	
Quality Technicians	S. Hicks								W. Sullivan								B. Jackson								
	J. Myers																								
Sick																									
Vacation	B. Allen																								

		MAINTENANCE																							
Week of	12/13/99	1st	T	W	Th	F	S	S	2nd	M	T	W	Th	F	S	S	3rd	M	T	W	Th	F	S	S	
Maintenance		D. Dyehouse							J. Williams								B. Krause								
		H. Winnale															K. Laubach								
		N. Hilley																							
		R. Pike																							
		B. Edens																							
Oiler		D. Lawwill																							
Janitor		C. Shearer																							
Sick																									
Vacation																									

(See Next Page for Overtime Assignments)

Figure 3: Gluing Schedule

GLUING DEPARTMENT/ WEEK OF 5-5-97										* Revised 5-2-97										M T W T F S S									
LEAD PERSON: F. Bates																													
NATORS:																													
12 HOURS:																													
A. Spicer																													
JA-PACKERS:																													
L. Lyle																													
W. Day																													
C. Edwards																													
J. Gahard																													
M. Meese																													
E. James																													
M. Ely																													
A. Fitzpatrick																													
T. Fleming																													
NEW HERE:																													
L. Gilbert																													
PALLETIZER/																													
CONVEYOR																													
G. House																													
SOR																													
CORE MTL HDL																													
12 hours																													
PTO:																													
12 hours																													
G. Osborne - LP																													
R. Rhodes																													
D. Wilson																													
SHIPPING:																													
F. Major																													
SICK:																													
R. Schwab																													
T. Bowler																													
VACATION:																													
F. Utene																													
C. Jackson																													
LAID OFF:																													
Schedule for																													
12 hours																													
TRADES:																													
7AM - 7PM Mon, Tues, Wed, Thurs, Fri																													
Sun 11PM - 7AM Mon - Thurs 7PM - 7AM																													
Fri 7PM - 11PM																													
Came in late																													
Left early																													

Figure 4: Litho Schedule

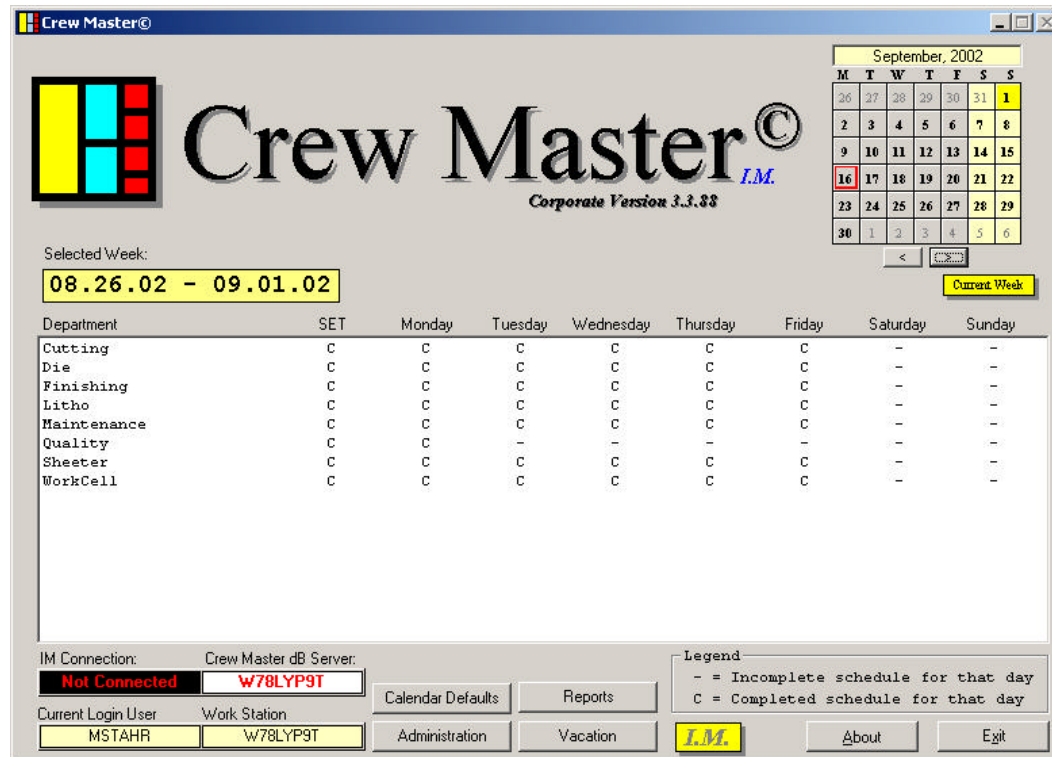
LITHO SCHEDULE WEEK OF 7-4-74 REVISED 7-1-74			
	1ST	2ND	3RD
100	C. VAN T. FYE A. HESS	D. HOWELL C. WELLS B. KING	D. POFFENBARGER J. BORDERS D. VIA
150	M. GREENFIELD N. STEWART G. RUSSELL	C. HOLLON T. HATHAWAY J. LENMAN	R. CHILDERS F. GILDFIELD J. SMITH
175	T. CARTER D. GLENN R. MILLER	S. DAY K. HICKS J. LINDSAY	B. BANKS T. PRESSLER E. GARRISON
LEADMEN	H. CARTER	R. PHILLIPS	
PLATEROOM	B. ALLEN T. RIVIZZINO D. HINKLE (U)	B. DOWLING R. PERRY	***** *****
AERIATORS	A. TRUESDELL	G. HINES	D. LAWILL
VACATIONS	C. CALDWELL C. MURRAY D. WINKLER T. MCCOY E. MURRAY J. SCHULER	(LAID OFF) NONE	
SICK	N. VANOVER		

Appendix C:  
Applications and Tasks



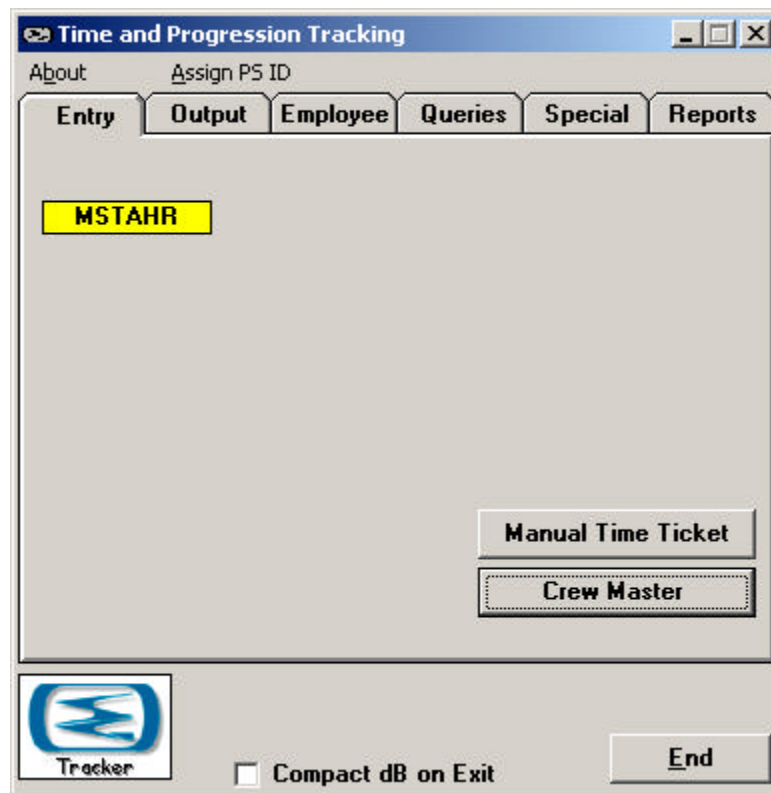
## 1. Crew Master

- ✍ Crew Master was developed to produce manning schedules for an industrial plant environment. The system allows users to produce weekly work schedules as well as vacation scheduling.



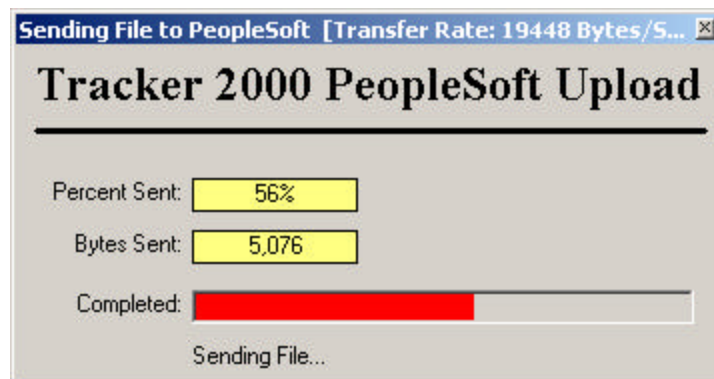
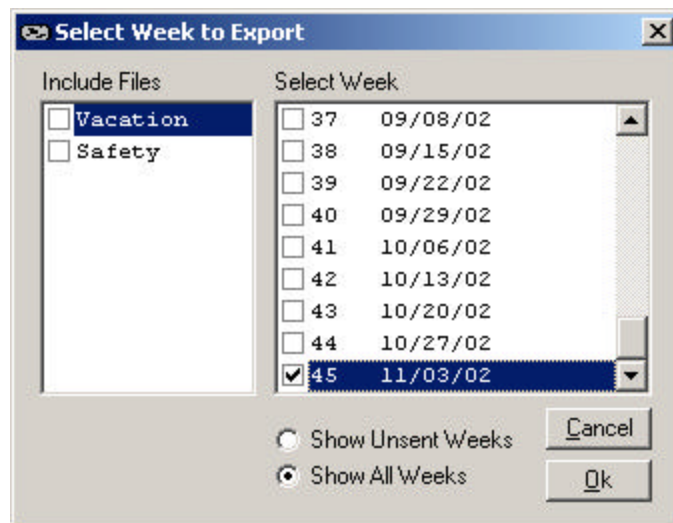
## 2. Tracker

✍ Tracker is an employee time and progression tracking system. The main purpose of the system is to track employee time to the level of machine positions. Tracker records hours per position for every employee hour worked. This collection can be input in various ways: 1) Traditional time ticket entry, 2) PC Pay import file, and 3) Kronos® import file. A weekly payroll file is generated and sent, via FTP, from Tracker to the corporate PeopleSoft® server.



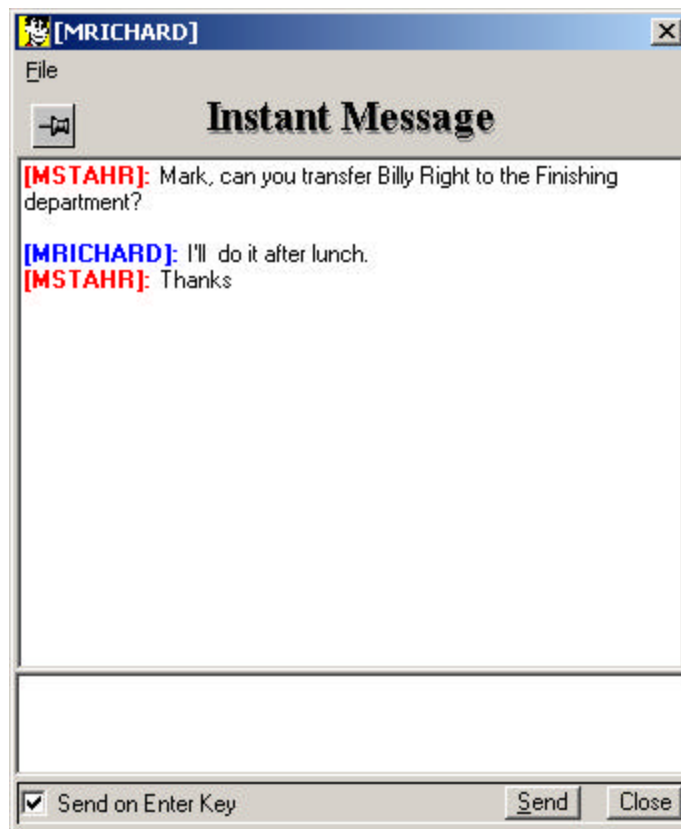
### 3. FTP Export

✍ Employee time information is atomically uploaded to PeopleSoft® via FTP. Each plant desiring to upload their time records directly to PeopleSoft® was asked to use a third party software called Chameleon®, however, I chose to create my own version of an FTP client in order to keep control of the process in-house.



#### 4. IM Server and ActiveX Control

✍ A server application was created in order to handle all communication between applications that incorporated the IM ActiveX control. The following screen shot is an example of the IM ActiveX control being used in Crew Master. This control was created in order to communicate between applications. In the example below the control is being used to send messages between users of Crew Master.



## 5. ActiveX Vacation Calendar

✍ The creation of an ActiveX vacation calendar is used throughout Crew Master. Its function is to incorporate the company business logic for employee vacation scheduling into a single control. The following is an example of one control that has been cloned as a control array in order to represent an entire year at one time. Vacation that is scheduled but not paid show up yellow. Once vacation is paid through Tracker the color of the scheduled vacation becomes green. White blocks represent company holidays and a red outlined day indicates the current day of year.

**2002 Vacation Scheduler**

Department: Finishing  
Employee: Edwards, Carol

Vacation Listing (Double-click to remove):  
04/01/2002-04/07/2002  
07/29/2002-08/04/2002  
08/05/2002-08/11/2002  
10/21/2002-10/27/2002  
12/02/2002-12/09/2002

Pay Only Listing (Double-click to remove):  
Monday Date    Hours    Paid?

Trigger Date: NA

Employee/Vacation Status:  
Allocated: 30 Days, 240 Hours  
Total Vacation: 30 Days, 240 Hours  
Pay Only Allowed: 30/30 Days, 240/240 Hours  
Scheduled/Used: 25 Days, 200 Hours  
Vacation Used: 0 Days, 0 Hours  
Pay Only Used: 0 Days, 0 Hours  
Remaining Vacation: 5 Days, 40 Hours  
Vacation: 5 Days, 40 Hours  
Pay Only: 5/5 Days, 40/40 Hours  
Pre Trigger: NA, NA  
Post Trigger: NA, NA


Legend:  
Green = Vacation has been paid  
Yellow = Scheduled Vacation  
White = Holiday

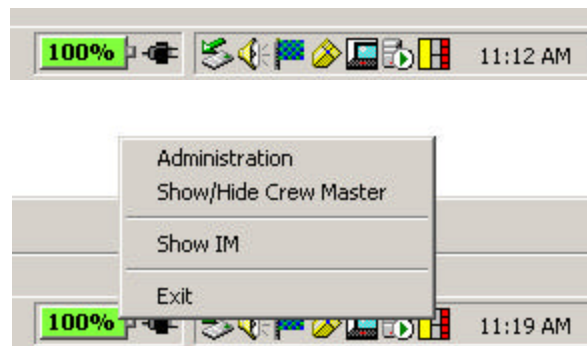
Selection Options:  
Select by Day  
Select by Week

Vacation Option:  
Schedule Vacation  
Schedule Pay Only

Buttons: Print, Confirm Vacation Removal, Overview, Unused, Close

## 6. System Tray ActiveX Control

✍ System tray icons are not a necessity but do offer a means of program control for the user. Menus can be displayed via tray icons which can help a user manipulate the application. The following two screen shots demonstrate the Crew Master Tray icon (  ) and one of the menu that is displayed when the icon is clicked by the mouse.



## 7. 3D ActiveX Button Control

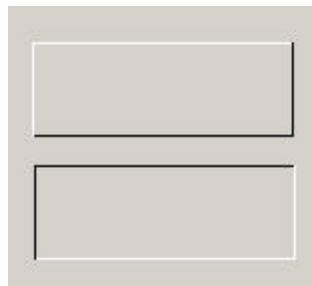
✍ I created a 3D ActiveX button control in order to give Crew Master a unique look. This control has no real effect on the workings of the system but does add an individual look to the system. The following screen shot displays an example of the 3D effect and unique look to the control.



## 8. ActiveX Border Control


✍ This ActiveX control is used as a container control for form level objects.

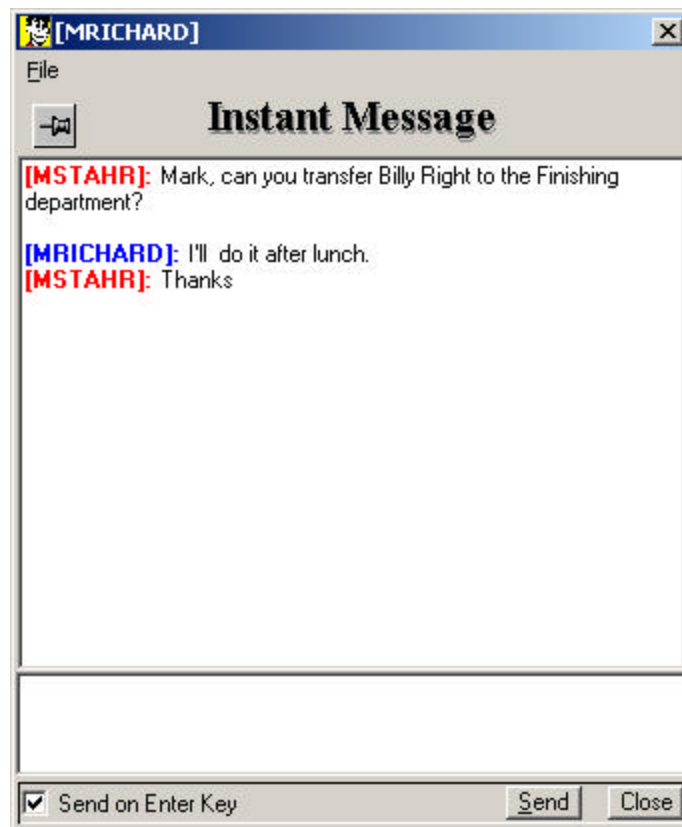
I added a property (BorderStyle) in order to give two distinct looks to the control. There is no functionality behind the control itself but rather a means to contain other controls. This control's visibility can be set to True or False which allows a programmer to quickly hide any objects that are contained in the control's border.





## 9. On Top ActiveX Control

- ✍ The On Top ActiveX control (  ) is also used on the IM screen. This control, when in its selected state, keeps the current window on top of every form in the Windows environment.



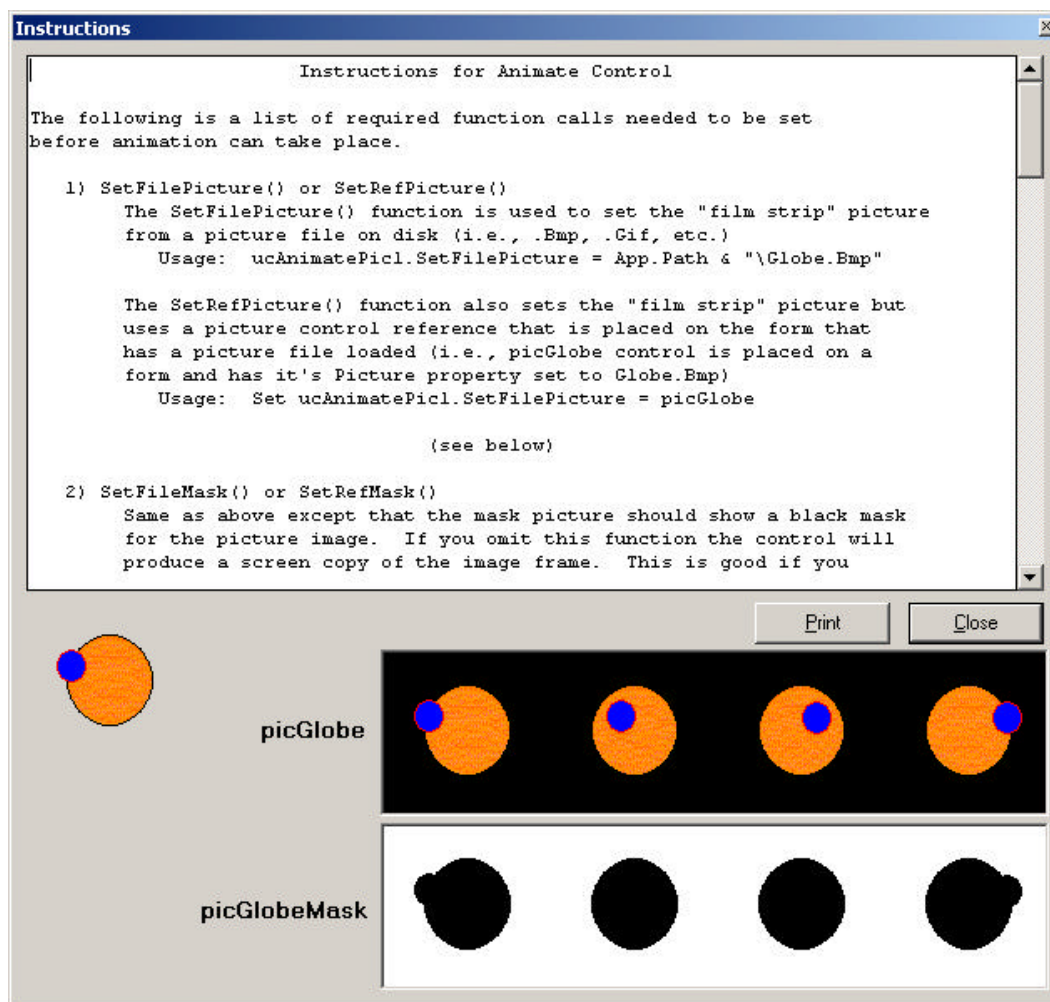
## 10. Login Info. ActiveX Control

✍ The Login Info ActiveX control is used as a quick reference to the active user's Windows NT username and the current machine they are logged onto. This functionality is also incorporated into the Custom Functions DLL file (see bulleted item 13 below) but this ActiveX control can be simply placed on a form's window without having to do anything coding. This control, once placed on a form, automatically reads the user's Windows NT logon name and machine name and places this data in the two appropriate labels. If a developer chooses to use the Custom Functions DLL to produce the same output they would need to place two labels on a form and manually set the caption property of each to the appropriate DLL function call (i.e., lblUserName.Caption = CustomFunctions.UserName)



## 11. AnimatePic ActiveX Control

✍ I created an animation control in order to have the flexibility to display graphics to the user. This control actually does nothing directly for the application but does offer a means of graphic animation for the system. One area this control can be utilized is in a “walk-through” demonstration. Crew Master and tracker both use the control but only in the About window.



## 12. Custom Functions DLL

✍ I found that I needed many of the same functions from one application to another. Retrieving a user's Windows NT logon name is one example. Because of this need I determined it would be beneficial to encapsulate these common functions into one file. The most logical solution was to create a Windows dynamic linked library (DLL) with each of the functions I would need. The following is a list of the functions that reside in this common function DLL file.

- Public Sub BreakRGB()
- Public Function BroadcastMessage() As Boolean
- Public Function CheckWindowsUser() As Boolean
- Public Sub ComboBoxSetHeight
- Public Function ComputerName() As String
- Public Function DaysInMonth() As Integer
- Public Sub EnableStartBar()
- Public Function GetComboBoxHeight() As Long
- Public Function GetDataSources() As String
- Public Sub GetDesktopRectt()
- Public Function GetListBoxHeight() As Long
- Public Function IfNull() As Variant
- Public Sub ListBoxLookUp()
- Public Function MikesEncrypt() As String

- Public Function MondayDate() As Date
- Public Sub MoveFileToRecycleBin()
- Public Sub Remove\_X()
- Public Sub SelectText()
- Public Sub SetListBoxHeight()
- Public Function SetListBoxTabs() As Boolean
- Public Property Set SetScreen()
- Public Sub ShowDesktopIcons()
- Public Sub ShowStartBar()
- Public Function SundayDate() As Date
- Public Function Swap() As Boolean
- Public Function SystemErrorDescription() As String
- Property Get Thinking() As Boolean
- Property Let Thinking()
- Public Function TrimOffEndOfString() As String
- Public Function UserName() As String

### 13. Error Logger DLL

✍ The ErrorLogger.DLL file serves as a generic component that can be included into any Windows development environment project (i.e., VC++, VB6, .Net, etc.) that will display, track, and save program errors used to help debug applications. I created this component in order to expedite code fixes while Crew Master is running in production. The following screen shot illustrates an example of the information this DLL supplies (I “hard coded” a “Division by zero” error to produce this shot). The caption of the window displays the path where the error log file is being saved (it is an option during setup to display this path to the user.) The component records the Windows error number, error description, the module and procedure where the error occurred, the line number in code where the error occurred, the computer name, the Windows NT user name, and text from the user explaining what they were doing at the time of the crash.

Creating this component proved to be extremely helpful during the initial release of Crew Master. I was able to track and fix bugs almost as fast as they were being reported. I have used other methods in the past which were less helpful to me such as a Dr. Watson report.

This DLL has recently been adopted as a corporate standard for internal developers creating Windows applications.

**Error Logging - [Path: C:\VB Projects\Smurfit\CrewMaster\_New\]**

Application: CrewMaster

Error Source: CrewMaster

Error Number: 11

Error Description: Division by zero

Module: modGeneral

Procedure: LoadDefaults

Line Number: 399

Please describe what you were doing at the time of the crash:

Computer: W78LYP9T

User: MSTAHR

Ok

Appendix D:  
Application Code Statistics



Application	Components	Controls	Procedures	Events	Methods	Properties	Lines
Crew Master	88	1065	1060	361	290	165	20,373
Tracker	110	1693	1410	558	326	193	31,879
FTP Import	7	8	130	7	26	43	1,254
IM Server	2	36	22	20	0	0	247
IM Control	5	55	37	26	8	0	824
Vacation Control	6	46	92	7	25	22	1,313
Sys Tray Control	3	6	21	3	5	3	310
3D Button Control	1	1	41	8	1	15	346
Border Control	1	11	0	5	0	1	74
On Top Control	2	12	28	13	3	2	304
Login Info Control	1	2	25	15	0	3	204
AnimatePic Control	4	9	51	7	12	15	688
Custom Functions DLL	1	0	31	0	27	2	727
Error Logger DLL	3	0	23	0	5	5	310
<b>Total</b>	<b>234</b>	<b>2,944</b>	<b>2,971</b>	<b>1,030</b>	<b>728</b>	<b>469</b>	<b>58,853</b>

Appendix E:  
Database Details

***A. Crew Master Tables (29)***

- 1. tbl\_AbsenteeReportHistory**
- 2. tbl\_Administrators**
- 3. tbl\_DailyScheduleComments**
- 4. tbl\_Departments**
- 5. tbl\_DryDockReasons**
- 6. tbl\_DryWetHistory**
- 7. tbl\_EmployeeGroups**
- 8. tbl\_EmployeeLocation**
- 9. tbl\_Employees**
- 10. tbl\_FTP\_EMS\_DepMap**
- 11. tbl\_FTP\_Import**
- 12. tbl\_HolidaySchedule**
- 13. tbl\_LaborGrades**
- 14. tbl\_MachinePositionLaborGradeLink**
- 15. tbl\_MachinePositions**

16. **tbl\_Machines**
17. **tbl\_PartTimeTransfer**
18. **tbl\_Print\_TransferHistoryPrint**
19. **tbl\_Schedule**
20. **tbl\_SchedulersRequirements**
21. **tbl\_Shifts**
22. **tbl\_TaskLog**
23. **tbl\_TransferHistoryLog**
24. **tbl\_Unions**
25. **tbl\_Users**
26. **tbl\_VacationAllocationStatus**
27. **tbl\_VacationRate**
28. **tbl\_VacationVerification**
29. **tbl\_Weeks**

***B. Crew Master Stored procedures (95)***

1. **spAbsenteeListing**

2. **spAbsenteeListingCount**
3. **spAbsenteeReportRecord**
4. **spAdd\_FTP\_Record**
5. **spAddDailyScheduleComments**
6. **spAddEmployee**
7. **spAddEmployeeGroup**
8. **spAddNewWeek**
9. **spAddSchedulersRequirements**
10. **spAddSplitShiftHours**
11. **spAddVacation**
12. **spAddVacationAllocation**
13. **spAdminPassword**
14. **spAllSplitShiftEmployees**
15. **spCheckSplitShiftRequest**
16. **spCopyEmployeeLocations**
17. **spCopyWeek**

- 18. spDeleteAbsenteeReportHistory**
- 19. spDeleteDayFromSchedule**
- 20. spDeleteEmployeeGroup**
- 21. spDeleteMachinePositionsFromSchedule**
- 22. spDeleteWeekFromDB**
- 23. spDepartments**
- 24. spDetailSplitShiftHours**
- 25. spDryDockReasons**
- 26. spEmployeeListForWeekAndDay**
- 27. spEmployeeLocate**
- 28. spEmployees**
- 29. spEmployeesOnVacationForWeek**
- 30. spFixOffset**
- 31. spGenericDepartmentVacations**
- 32. spGetAllWeeks**
- 33. spGetDailyComments**

- 34. **spGetDep\_Mach\_Pos**
- 35. **spGetDepartmentCompletedDayStatus**
- 36. **spGetEmployeeGroups**
- 37. **spGetEmployeeScheduledDepartment**
- 38. **spGetEmployeesToRemoveFromMachPos**
- 39. **spGetFTP\_EMS\_DepMap**
- 40. **spGetLastSchedulersEntry**
- 41. **spGetListOfEmployeesOnVacation**
- 42. **spGetRowsToFixOffset**
- 43. **spGetSchedulersRequirements**
- 44. **spGetSplitShiftTransEmps**
- 45. **spGetUniqueTaskLogDates**
- 46. **spGetVacationKey**
- 47. **spGetWeeksToSynchronize**
- 48. **spHolidaySchedule**
- 49. **spInsertDryWetEmployee**

- 50. spInsertEmployeeLocation**
- 51. spInsertFTP\_EMS\_DepMap**
- 52. spInsertScheduleRow**
- 53. spIsWeekPosted**
- 54. spLoadDaySchedule**
- 55. spLoadDryWetEmployees**
- 56. spLoadVacationAllocation**
- 57. spLogTask**
- 58. spMachinePositions**
- 59. spMachines**
- 60. spMaxWeekPosted**
- 61. spNewAndOldMachinePositions**
- 62. spRemoveEmployeeFromSchedule**
- 63. spRemoveEmployeeFromSplitShift**
- 64. spRemoveFromScheduleAddToDryWet**
- 65. spRemoveTask**



- 66. spRemoveTasks**
- 67. spRemoveVacation**
- 68. spSaveAbsenteeReportHistory**
- 69. spSaveDaySchedule**
- 70. spScheduledVacations**
- 71. spSetNewFTPLoad**
- 72. spSETVacationDefaults**
- 73. spShifts**
- 74. spSyncEmployees**
- 75. spTasks**
- 76. spTransferEmployee**
- 77. spTransferHistoryLog**
- 78. spUnions**
- 79. spUnscheduledVacationAllocation**
- 80. spUnusedVacations**
- 81. spUpdateContinuousRunCycle**

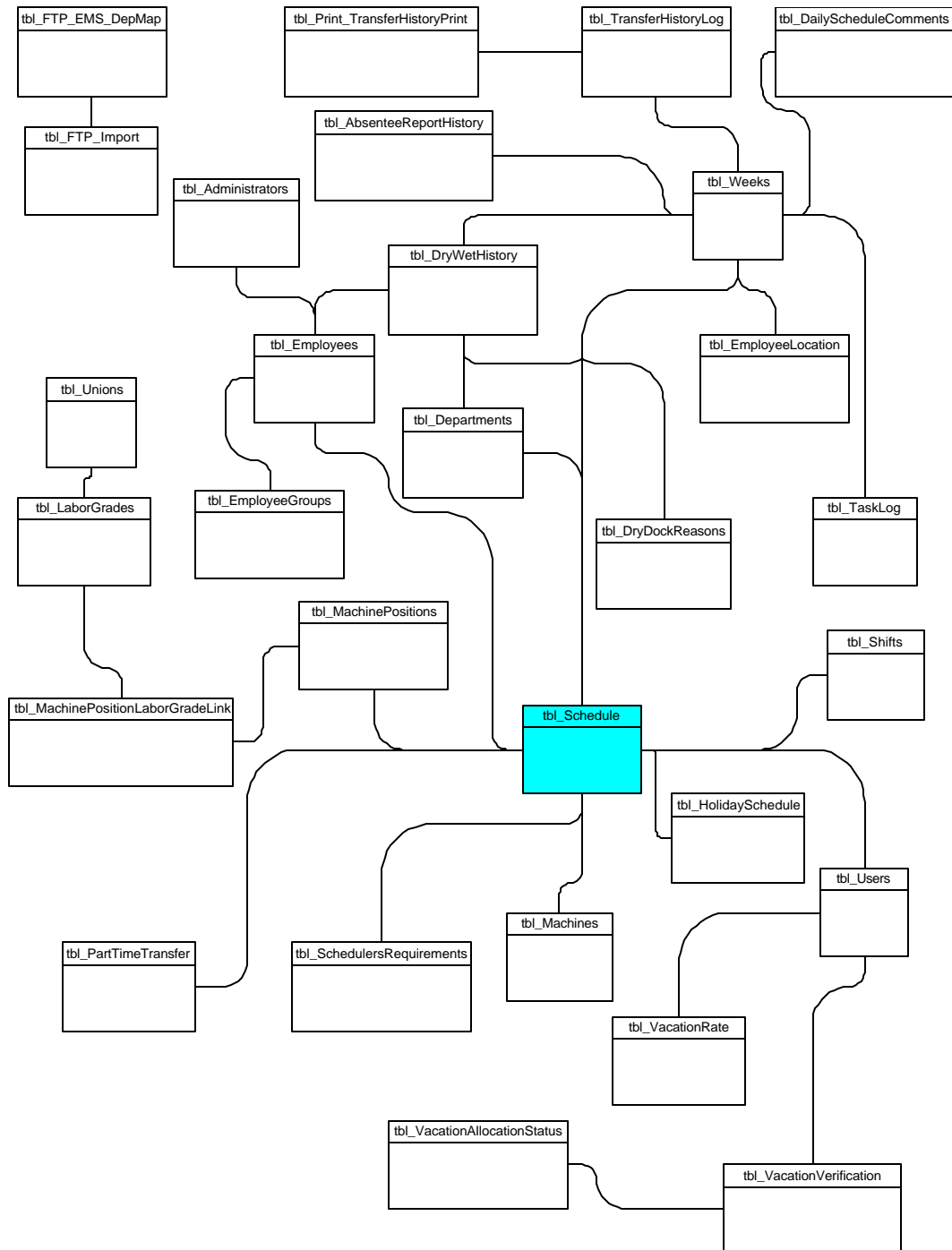
- 82. spUpdateEmployee**
- 83. spUpdateEmployeeFromFTP**
- 84. spUpdateEmployeeGroup**
- 85. spUpdateEmployeeStatus**
- 86. spUpdateEmployeeVacationAllocation**
- 87. spUpdateFTP\_EMS\_DepMap**
- 88. spUpdateMachPosWithInactiveMachines**
- 89. spUpdateSplitShiftHours**
- 90. spVacationByGroup**
- 91. spVacationDefaults**
- 92. spVacationEmployees**
- 93. spValidUser**
- 94. spVerifyEmployeeLocation**
- 95. spWeeks**

***C. Crew Master Views (2)***

- 1. vw\_UpdateEmpFromFTP**

## 2. vw\_UpdateEmployeeDepFromFTP

### D. Crew Master Table Diagram





Appendix F:

Visual System Outline Data Flow

