Plus Chart Process Modeling Software

Erin Handberg
Stephen Fjelstad
Joshua Marcoe
Eric Walz

Math and Computer Science Department
South Dakota School of Mines and Technology
Rapid City, South Dakota 57701
handberg@rushmore.com

Abstract

The Plus Chart Process is a new modeling tool which encapsulates the power of other commonly used design tools while adding an elegantly simple way to organize and view an entire project. Development of a software package to implement the Plus Chart Process will allow the users to translate their thoughts from natural language to the Plus Chart model, improving the practicality of process development and design.

Software design for the graphical user interface brought the Plus Chart Process to life by integrating the process steps with an easily manipulated visual model. Users can construct models of a procedure, perform trade studies by developing alternate paths, perform data mining, and determine the best path for the process they designed prior to physical implementation. This paper discusses the implementation of the software package, the successful results of applying the process, and the lessons learned in applying the Plus Chart Process.

1 Introduction

Well-designed process models result in structured, organized projects which require little rework late in the lifecycle. Therefore, a complete and powerful process modeling tool becomes a necessity in achieving a stable design. The Plus Chart Process retains the power of commonly used design tools while adding an elegantly simple way to organize and view the design process.

The Plus Chart Process provides users with a way of organizing steps in a visual way by defining each step as a plus symbol. Each plus symbol has a horizontal primary path, a vertical secondary path, and a functional step with optional control signals. The primary path contains the primary input, the functional step, and the primary output. Likewise, the secondary path contains the secondary input, the same functional step as used in the primary path, and the secondary output. A primary pathway can be modeled by assembling a combination of pluses. A project then becomes a collection of primary pathways that interact with each other via input links, output links and control signals. The control signals may initiate the start or signal the end of a functional step to regulate timing between individual pathways. Projects are then visually displayed using plus symbols linked together and can be changed easily to find the optimal arrangement of steps.

The development of a simulated assembly line demonstrates the Plus Chart Process applied to a real-life project. The application focused on achieving an understanding of all possible combinations of the PCP, then testing each scenario by applying it to the assembly line. The project resulted in an optimal assembly line arrangement within the project constraints.

The senior design team, SE^2, consisting of Eric Walz, Stephen Fjelstad, Joshua Marcoe and Erin Handberg, developed a software package using the Plus Chart Process following a combination of systems engineering and object-oriented design methods. Early interviews with the Plus Chart Process creator, Kevin W. Patrick [1], and an in-depth study of his white paper formed the foundation for operational concepts and requirements development. Detailed design of the software matured when the top-down development incorporated the Plus Chart Process with object-oriented methodologies while being applied to the simulated assembly line. Software implementation advanced with the application of the Plus Chart Process to the simulated assembly line, streamlining the procedure and increasing the efficiency of the process.

2 Plus Chart Process

The *Plus Chart Process* (PCP) provides users with a way of organizing sequential steps; therefore, any logical process can be modeled using the Plus Chart. The visual ideogram inherent in the PCP illustrates the natural progression of the process. Each plus includes a horizontal primary path, a vertical secondary path, and a functional step with optional

control signals. The primary path functions as the value added line or the main assembly line. The secondary path contains the overhead and support line. The functional step sits in the center of the two lines, combining the inputs and producing the output for each path.

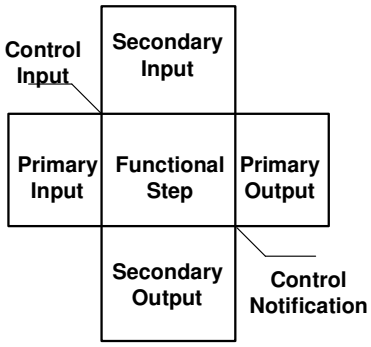

Figure 1: Plus Chart Process Element

As shown in Figure 1, and by delving a little deeper into each of the paths in the process, each path needs specific information in the input and output steps to complete the path as designed. The Primary Input step contains information needed in the main assembly line of the system. For example, this information could include value-added parts, specialized equipment, quality assurance check steps, and so on. The Secondary Input provides information for the support line of the system. This support inputs include personnel, accompanying materials, equipment used, special instructions and other documentation.

The Functional Step contains the task to be performed within the plus. The task is expected to be a single step in a process. The step can be viewed from a very high-level and encompass a lot of smaller steps as a overview, or the step can be detailed and specific, such as checking spark plug spacing. In either case, the functional step needs parts, personnel, and tools for the task to be completed, and these items come from the Primary and Secondary Input blocks. After the actions have been performed in the functional step, personnel and materials not needed later in the process are removed from the process via the Secondary Output. The completed materials, personnel and equipment intended to progress in the process become part of the Primary Output.

In addition to the five main blocks already discussed, the Plus Chart has two optional control flags which can be used in signaling other processes. The Control Notification flag and the Control Input flag work in tandem. At the completion of a functional step, the Control Notification flag associated with that step signals its matching Control Input flag. When the Control Input flag receives the signal, the functional step it is associated with may begin its task. Because this process is difficult to see at this point, we will be discussing the control flags further as the process evolves.

Now that we know how a single plus is built, we need to look at linking pluses together in a specific sequence. A primary pathway can be modeled by assembling a combination of pluses. If we have two pluses, A and B, the output of A will need to match the input of

B before the two pluses can be linked. After the two pluses are linked, the overall process will contain both tasks. First, task A will be performed, the output of A will pass to the input of B, and then task B will be performed. Users continue to link pluses together in this way until the process contains all of the necessary steps. In this way, a primary pathway is a collection of pluses with a specific starting point and a specific ending point.

A project becomes a collection of uniquely named primary pathways that interact with each other via input links, output links and control signals. A single primary pathway, referred to as the *Main Assembly Path* (MAP), is generally considered the primary path of the system. Additional primary pathways called Subassembly Paths provide input to the MAP via secondary inputs. Control signals may initiate the start or signal the end of a functional step to regulate timing between individual pathways. For example, the completion of a step on the MAP can trigger the start of a Subassembly Path using the control signals. If the timing is set up correctly, the Subassembly Path will provide its product to the MAP without delaying the MAP. This is referred to as *Just-In-Time* (JIT) processing, as shown in Figure 2.

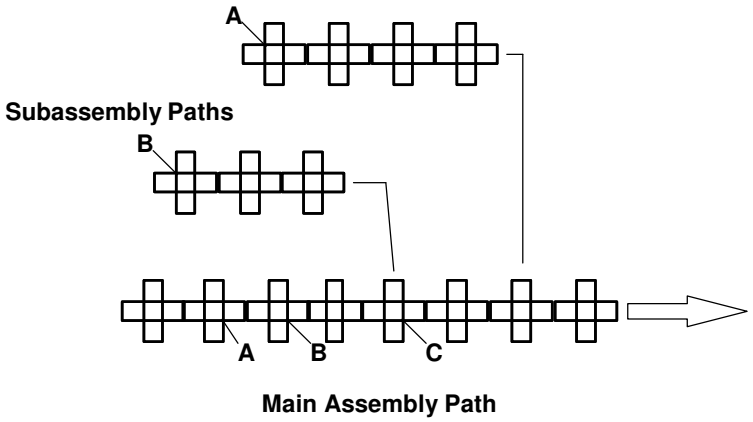

Figure 2: Just-In-Time Processing

The Plus Chart Process allows users to consolidate a string of pluses in a given path into one generic plus to better facilitate an overview of the process, as shown in Figure 3. The consolidated plus can be expanded to show a hierarchical structure. This ability is used frequently throughout the system to illustrate the broad, overarching nature of the model.

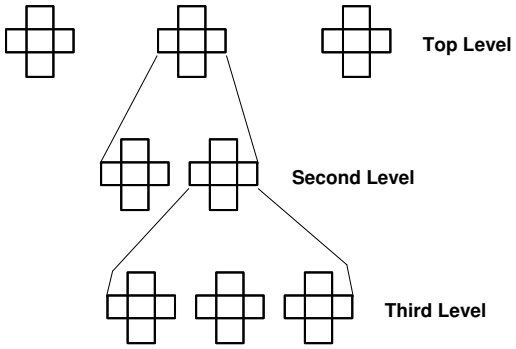

Figure 3: Hierarchy Paths

In addition to allowing Subassembly Paths and hierarchy, the Plus Chart Process also allows certain tasks to be completed in parallel, as shown in Figure 4. For example, if one segment of the application can be done at the same time as another segment, we allow the users to make that part of the process parallel. One parallelized pathway stays as part of the MAP. The other parallel pathways branch from the MAP and create a sub-path which completes the steps in parallel.

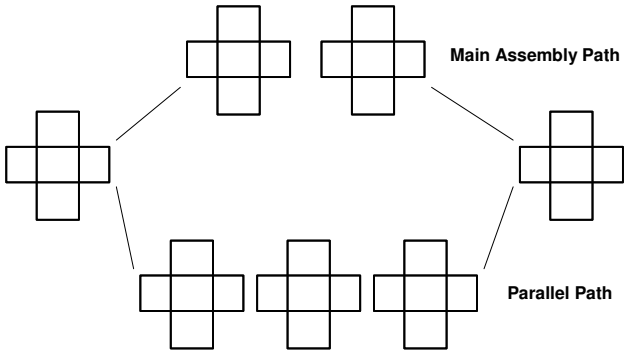

Figure 4: Parallelizing the Process

After developing the Plus Chart Process model for a particular assembly line, the flexibility of the Plus Chart allows users to analyze other options to optimize the model. Case study analysis can be performed by developing an *Alternate Path*, which is a collection of pluses forming a separate primary pathway. This separate pathway can be used to replace a series of steps in the MAP. Because the end result of the *Alternate Path* is identical to the output of the series replaced, the effect on the system is determined by the unique characteristics of the pluses on the *Alternate Path* in comparison with the MAP. *Alternate Paths* provide users with an option to perform detailed analysis of the system to find the best configuration possible, as shown in Figure 5.

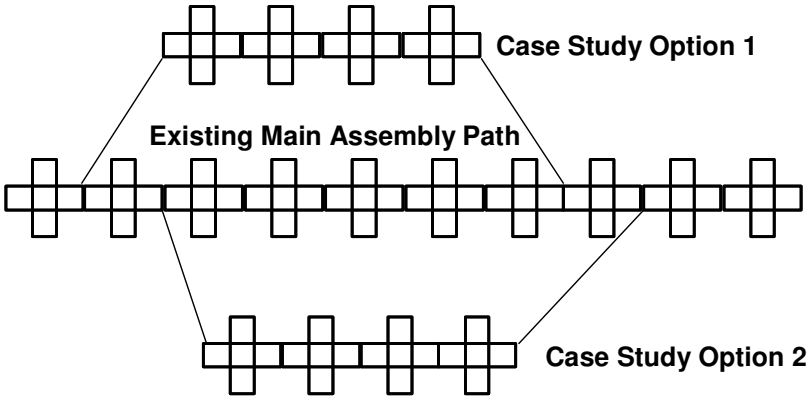

Figure 5: Case Study Example

The Plus Chart Process provides users with an effective way to model and optimize a system. Using parallelization, hierarchy, subassembly paths and case studies, users can

efficiently illustrate a process and its alternatives while maintaining an overview of the system as a whole.

2.1 Simulated Assembly Line

The power of the Plus Chart Process can be demonstrated by applying it to an assembly line. The assembly line chosen for this description restores a 1956 Chevy Bel Air. In order to apply the Plus Chart Process to this application, the steps in the restoration must be identified. The restoration can be described in extreme detail; however, a high-level overview of the assembly line will still demonstrate the power of the process.

To restore a 1956 Chevy:

1. Begin Disassembly:
    a. Engine
    b. Body
2. Media Blast the body
3. Repair defects on the body:
    a. Cut out rust
    b. Fabricate patch panel
    c. Weld in patch
    d. Grind smooth
    e. Repeat
4. Paint the body:
    a. Smooth rough edges
    b. Sand
    c. Prime
    d. Paint
    e. Clear coat
5. Apply graphics to body
6. Remove Engine from Chassis
7. Disassemble Engine
8. Rebuild Engine:
    a. Build basic engine
    b. Add fuel injection
    c. Add turbo
9. Powder coat Chassis
10. Mount Engine onto Chassis
11. Mount body onto Engine and Chassis
12. Design Interior
13. Create interior panels:
    a. Create front seats
    b. Create back seat
    c. Create dashboard
14. Install interior
15. Polish Car
16. Deliver Car

2.2 Modeling the Simulated Assembly Line

To simplify the use of the Plus Chart Process, SE^2 developed a procedure which allows the user to effectively model the assembly line and to ensure compliance with the PCP model. The following procedure describes applying the PCP to an assembly line:

- Build single plus to determine inputs and outputs of the system
- Break the overall plus into a single path of individual pluses
- Determine dormant inputs for parallelization
- Parallelize the MAP
- Determine Subassembly Paths
- Break Subassembly Paths out of the MAP
- Link Subassemblies into MAP
- Determine sections of each path for consolidation
- Consolidate pluses to develop hierarchy
- Review the entire system with a data mining approach
- Determine *Alternate Path*s for case studies
- Evaluate case studies
- Determine the optimal system configuration
- Ensure model matches inputs and outputs of initial plus
- Repeat as long as necessary

To model the assembly line depicted in Section 2.1, the process begins by building a single plus for the overall system. The single plus describes the intended input and output of the process and will serve as a validation check later in the procedure. For the car restoration, this plus will have the old car for a primary input, a restored car for the primary output and a general statement for the secondary input and output encompassing all parts, personnel, and waste materials used or generated in the process.

Now that a starting point has been defined for the application, in this case the old car, the process needs to be broken down into a series of tasks. Each task will be represented by an individual plus that when linked describes the entire system. More specifically, the old car becomes the primary input for the first plus and the starting point for creating the linear system. The process will link each successive plus into the MAP, where the primary outputs of each plus match the primary input of the following plus. The outcome of this process results in the newly restored car. The construction of the MAP can be illustrated by reading the numerical headings, and ignoring the alphabetical subheadings, in the outlined form shown in Section 2.1, as shown in Figure 6. Developing the hierarchical structure with the alphabetical subheadings will be discussed later.

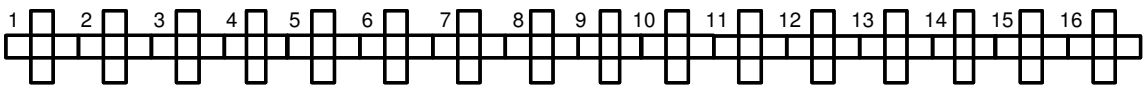

Figure 6: Initial Main Assembly Path

While the MAP can encompass the given example and allow for a valid simulation, the PCP allows the user to identify areas of the process that can be manipulated to better

optimize the system. Parallelization, *Alternate Path*s, and hierarchy are examples of areas the PCP can manipulate the system.

In the next step of the procedure, the PCP assists the user in identifying potential parallel applications. To parallelize the system, the user must first locate *dormant* inputs. A dormant input is an item or group of items which travel through many pluses in a path. These inputs are not acted upon in the functional steps until later in the system.

Dormant items in the path highlight potential parallelization opportunities which speed up the overall application process. In this example, the engine and chassis remain as dormant inputs until the body is completed. Similarly, the body remains dormant while the engine and chassis are rebuilt. Using the PCP, the MAP breaks into two separate paths and eliminates the dormant inputs thus increasing the optimization of the system.

The Plus Chart Process has helped identify ways to optimize the process, and to further illustrate this, the remainder of the section will be devoted to describing the car restoration example.

To parallelize the system from the MAP, the user must rebuild parts of the path to incorporate the parallelization. In this example, the first plus, referred to as *Begin Disassembly,* remains unchanged and starts with the old car as the primary input, people and tools as the secondary input, and begins the disassembly process as the functional step. The secondary outputs include scrap materials, tools no longer needed, and people who will not progress with the car to the next step. The primary output contains the body of the car, the engine and chassis, and all other people and materials which progress in the process.

The parallel processes identified earlier start from *Begin Disassembly*. For this example, the MAP progresses forward to build the body, while a second pathway referred to as the *Engine Path* branches in parallel to complete the engine and chassis modifications, as shown in Figure 4. The *Engine Path* will merge into the MAP later in the process. More specifically, the primary output of *Begin Disassembly* divides its resources between the first plus in the *Engine Path* and the next plus in the MAP.

The MAP and the *Engine Path* are developed until the parallel nature is no longer possible. At that point, the *Engine Path* is merged back into the MAP at the *Mount Body onto Engine and Chassis* plus.

In the original sequential series of the process, the interior development would not start until this point in the process. However, by identifying that the interior can be designed and constructed without the use of the body or chassis, an *Interior* Subassembly Path can be created to perform these tasks separately from the main process. The primary output of this Subassembly Path will feed into the *Install Interior* step as a secondary input.

The design and creation of the interior would not take the same amount of time as the rebuilding of the chassis and the motor. However, if we start the *Interior* Subassembly

Path at the same time as the disassembly of the old car, the completed interior will remain dormant until the rest of the car is ready for the interior installation. Setting a control notification flag on the *Paint Body* step and the matching control input on the *Design Interior* step the interior will be ready in a *Just-In-Time* (JIT) manner. This eliminates the wait time of the interior and streamlines the process.

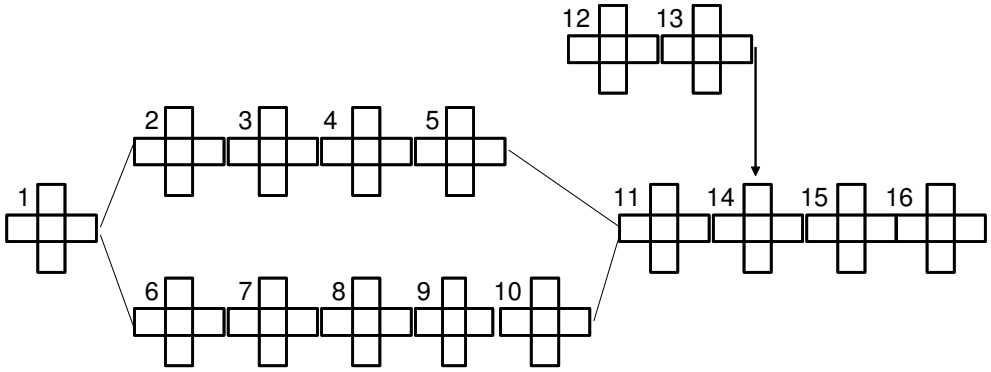

Figure 7: Parallelized Car Restoration Assembly Line with Interior Subassembly

The parallelization and streamlining of the assembly line reduced the overall time needed to restore the car, and the PCP provided the user with a visual overview of the assembly line, as shown in Figure 7.

Even though the car restoration has been improved drastically by adding parallelization and subassembly paths, the alphabetical subheadings in the outline have not yet been addressed. These steps can be easily incorporated by adding hierarchy into the system. A hierarchical structure can be developed by combining a sequence of pluses into a single overarching plus. This process has already been illustrated in the outline. For example, the process steps of *Build Basic Engine*, *Add Fuel Injection* and *Add Turbo* can be described in the single step of *Rebuild Engine*, as shown in Figure 8. Adding hierarchy better illustrates the system in an overall setting, as well as allowing users to drill down to minute details in the system.

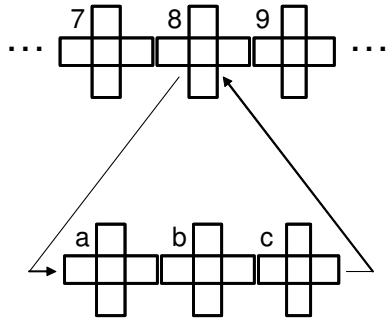

Figure 8: Hierarchical Structure for Task 8

Up to this point, the optimization of this procedure has merely entailed adding parallelization and developing subassembly paths. System analysis using a data mining approach can be performed to fully understand all consequences of the system. System

analysis is limited only by the detail entered into the plus and the determination of the user to acquire the information. In addition to system analysis through data mining, further optimization can be performed by developing *Alternate Path*s and performing case studies. Case study analysis allows users to reconfigure the system and determine if that reconfiguration will provide a more optimal result. Needless to say, data mining and case study analysis can become computation intensive and will not be described in detail.

This process is fairly time-consuming to complete manually as illustrated here. As the system becomes more detailed and complex, the effort users expend to analyze the optimal configuration increases almost exponentially. The process becomes even more tedious when evaluating case studies due to the replication of work developed by the system. Because of the time and effort involved in creating diagrams and difficulty in determining some of the data mining information, SE^2 chose to develop software to enhance the functionality of the model and to provide the users with a more automated way to find potential problems without requiring full knowledge of the PCP.

3 Software Development

3.1 Overview

To begin design of the software, SE^2 and Patrick worked together to determine high-level requirements for the *Plus Chart Modeling Program* (PCMP). This allowed the team to develop software requirements and apply the Plus Chart Process to the Simulated Assembly Line. After a thorough review of the requirements by Patrick, the team established the requirements as the foundation of the software development process, reviewed the document every two months, and ensured additional or changing project design met the requirements.

SE^2 chose to use a combination of object-oriented and traditional software engineering practices to organize the software development of the PCMP. One member of the team had experience in industry with traditional software engineering practices; however, the nature of the project demanded an object-oriented approach. Therefore, the synthesis of the two practices was used in an attempt to effectively organize the software design and team progress. Rather than describing the specific methodology used here, each part of the design will be described as either structured or object-oriented and the justification for using that technique will be provided in that section.

3.2 Requirements and Software Design

The Plus Chart Process is highly dependent on the data integrity of the system; therefore a strong database design became the most important aspect of the software design. Much of the database design work focused on understanding the scope of the project and the interactions of the entities in the Plus Chart. The database needed to support the structure of the Plus Chart in addition to providing the necessary data and flexibility to be used in

other modules of the program. Based on the current knowledge of the team, SE^2 developed a relational database which seemed to work well for the system. While an object-oriented database design may have served the project better, this option was not explored due to time constraints.

After completing the initial database design, the team began detailed design of the software. Detailed design consisted of breaking the project down into nine modules, as shown in Figure 9. Each of the modules will be described in the remainder of this section.

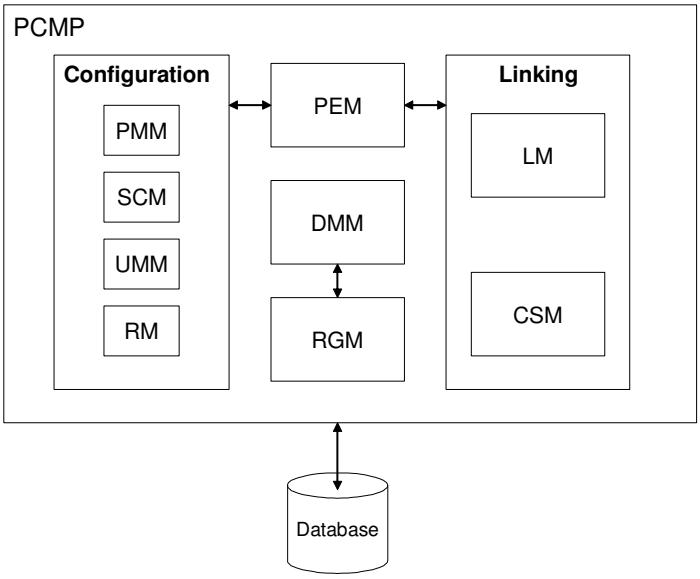

Figure 9: Software Architecture

The *Project Manipulation Module* (PMM), *System Configuration Module* (SCM), *User Manipulation Module* (UMM), and *Resources Module* (RM) were the first part of the software development. The PMM allows users to create, modify and delete projects used in the program. The SCM allows the program administrator to configure the PCMP for changes in the location and usage of the database. The UMM allows the users to create, modify, and delete users to the system. The UMM also allows users to determine the user level access for projects and infuses a small degree of security to the project. The RM enables users to create instances of all the materials and personnel that are going to be used in the project.

Once these modules were developed, the team started work on the *Process Element Module* (PEM). The PEM is perhaps the most important module in the program because it enables the user to create each individual plus in the process, and nothing can be completed in the design of a model beyond this point. Similarly, the PEM needed to be designed before any of the remaining modules could be fully developed.

The building process of each plus was broken down into five creation steps which culminated with a validation step to ensure compliance with the PCP. These creation steps include primary input, primary output, secondary input, secondary output, and

functional step. The design of the input and output steps allowed the user to search and select resources from a list of available resources and add the desired quantity. The functional step enabled the user to set the preliminary path for the plus and allowed the user to describe the activities happening inside of a given plus through a usage table. The usage table described the resource, the quantity used, and how the resource was used. This usage information ensures the quantity of the resources input, modified by the quantity of how the resource was used, results in the given output of a plus. This allows the PEM to perform a validation check for each plus.

Linking the pluses is done in the Linking Module (LM). The LM was broken down into four steps: primary linking, secondary linking, control signals, and hierarchy. Primary linking takes the pluses on a given preliminary path and allows the user to link the pluses in the order they desire. Secondary linking allows the user to take outputs from a Subassembly Path and link them with the corresponding secondary input of another path. The control signals section creates the JIT aspect of the PCP by creating send and receive signals in the process. Hierarchy is the final step in the linking process allowing the user to have a general plus generated which is the summation of a subset of pluses in a given path.

The Case Study Module (CSM) allows the user to create an alternative path of pluses that can be evaluated. This *Alternate Path* is constructed using the same methods as linking, but depicts a separate *optional* path.

Valuable data can be extracted from the system via the Data Mining Module (DMM). The DMM enables the user to select from a set of queries to determine the validity of the system and provide analysis. The data collected by the DMM is displayed using the Report Generation Module (RGM).

3.3 Implementing the Project

After completing the design of the project, SE^2 chose some of the specific implementation information for the project. Because the team wanted the software to have platform independence, they decided to implement the PCMP using Java 2 and a MySQL database. *Graphical User Interface* (GUI) design would be completed in Swing objects, and implementation would be performed in the Net Beans *Integrated Development Environment* (IDE).

The coding methodology used to develop the PCMP was based on object-oriented software engineering practices. Code reuse was a critical factor due to the similar functions of many modules in the program. General classes were developed for common functions of the program like database connections and queries. These classes, a prototype database script, and the responsibility of development were distributed to all team members. Once these modules were completed, the development of the PEM, LM, DMM, CSM and RGM was shared by team members and developed in parallel as much as possible.

With the majority of the system developed in parallel, the project required system integration and testing to pull all of the pieces together. The system integration was completed by the team as a group so the problems which arose could be addressed by the module creators.

4 Modeling the Simulated Assembly Line with the PCMP

Modeling the Simulated Assembly Line with the PCMP is a simple process. First, a user account is created. The user then creates a project. At this time, a user can begin designing pluses in the PEM or begin entering resources in the RM. Users will need to enter at least some resources prior to completing the first plus in the PEM; however, if all of the resources are not known at the onset of the project, they can be added anywhere during the development of the model.

After resources are entered into the system, the user needs to create the pluses (steps) of the process. Using the PEM, the primary inputs, primary outputs, functional step, secondary inputs and secondary outputs are created for each plus in the system. Each plus is assigned a path in the PEM and is validated to ensure consistency. Any anomalies are to be fixed by the user before proceeding.

The LM is used to link each plus in the correct order. First, the user selects pluses to be linked via primary input and primary output. Once each path is completed, separate paths are linked via secondary inputs and outputs, followed by adding control signals. If the user wishes to generalize a series of pluses for ease of at-a-glance review, the hierarchy portion of the LM would be used.

Once all paths are linked, the system can be considered valid; however, the user can use the CSM to develop a limited number of alternate cases to be studied. Essentially, the CSM creates a small series of pluses that have the same input and output as a similar series on the MAP. In the example above, an alternate case might be to paint the chassis instead of using a powder-coating. The user would create the pluses to paint the chassis using the PEM, link the pluses via primary inputs and outputs, and then use the CSM to identify this new *Alternate Path* as a case study path. Based on the user determined specifications, the DMM will then provide the user with a comparison of the *Alternate Path*s and identify the most optimal configuration.

The DMM can be run any time after the system has been linked and can be considered valid. The DMM prompts the user with a host of options ranging from determining cost to finding dormant inputs in the process. The results of the DMM operations are then displayed to the user using the RGM.

Based on the results displayed by the RGM, the system can be modified using the PEM and LM. Parallel paths are identified using the DMM to determine the dormant sections of the system, and modifying these sections to run in parallel.

As more modifications are made to the system, the optimal system will evolve. Using the PCMP to simulate the system, make modifications, and re-simulate the system prior to physical implementation, time and money can be saved.

5 Conclusions

The Plus Chart Process is a new modeling tool which is powerful, flexible, and provides the unique ability to optimize a system. The Plus Chart Process can be generalized for projects of all sizes, levels of detail, and complexity while maintaining integrity of the application. The versatility of the model allows effective adaptation to the most complex systems, efficiently illustrating a process and its alternatives while maintaining an overview of the system as a whole.

The only downfall to the Plus Chart Process is the amount of time and effort to convert a large-scale process into the model by hand. Realizing the time-consuming nature of the system, development of a software package reduced the overhead and streamlined the process. In addition, the development of a robust database, flexible data mining options, and easy comparisons using case studies, improved the accuracy and reliability of the system while reducing manual computation and calculation time. The introduction of a procedure to convert existing processes to the Plus Chart model allows an increase in user effectiveness without requiring users to understand the intricate details of the model.

The evolutionary shift of the Plus Chart Process to a software-based platform is a necessary step in utilizing the true power of the process. The desire to create a software package which maintains the flexibility of the PCP has enhanced the usability and efficiency of applying the process. As the software continues to evolve, advances in parallelization detection, case study modeling, and more accurate and detailed data mining solutions will revolutionize the PCP and perhaps cause a shift in modeling paradigms.

Appendix A

Acronym List

PCP	Plus Chart Process
MAP	Main Assembly Path
JIT	Just-In-Time
PCMP	Plus Chart Modeling Program
PMM	Project Manipulation Module
UMM	User Manipulation Module
SCM	System Configuration Module
RM	Resources Module
PEM	Process Element Module
LM	Linking Module
CSM	Case Study Module
DMM	Data Mining Module
RGM	Report Generation Module
GUI	Graphical User Interface
IDE	Integrated Development Environment
OOP	Object-Oriented Programming

References

1. Patrick, Kevin W. *Plus Chart Process © 2002.* Personal Communication. 2005.