

*Team 6: Mellow Yellow  
Project Plan*

CS 422  
9/14/99

Nick Gunder  
Kerry Hammil  
Azrina Hussin  
Ryoji Noda  
Brock Rogers  
Alex Velkov

## **Abstract**

This project plan summarizes our team's approach to the CSPN graphical user interface project. It describes:

- Roles of individual team members
- Functional description of the software
- Proposed enhancements to the software specification
- Development environment and procedures
- Perceived risks to the project
- Basic testing approach
- Tentative schedule for development and deliverables

With this plan and the procedures it describes in place, we are ready to begin on the project itself.

# Table of Contents

Abstract .....	ii
Table of Contents .....	iii
1. Introduction .....	1
1.1 Executive Summary .....	1
1.2 Goals .....	1
1.3 Definitions .....	1
1.4 Organization .....	1
2. Partitioning Strategy .....	3
3. Basic Functional Description .....	4
3.1 The Command Line Communication and Interpretation Section .....	4
3.2 Third Party Application and Format Support Section .....	4
3.3 Interaction Section .....	4
4. Development Environment .....	7
4.1 Software .....	7
4.2 Platform .....	7
4.3 Libraries .....	7
5. Risk Assessment .....	8
6. Testing Approach .....	9
6.1 General Strategies .....	9
6.2 Defect Testing .....	9
7. Development and Deliverables Schedule .....	11

## *List of Figures*

**Figure 1:** CSPN GUI Context Diagram

## *List of Tables*

**Table 1:** Development and Deliverables Schedule for CSPN GUI Project, Team Mellow Yellow

# 1. Introduction

## 1.1 Executive Summary

The Mellow Yellow software engineering team has been formed on 23 Aug 1999 with the purpose of building a GUI for the CSPN system. The GUI will enable the user to launch their favorite editor to create and modify files or use basic GUI editing capabilities. It will also support the display of text and graphic files, and provide context-sensitive help. The GUI must be able to interface smoothly with a separately implemented portion of the system that deals with the port. To accomplish these tasks, the Mellow Yellow team consists of two developers, two testers, and two documentation and specification experts. The project is slated for completion before 15 Dec 1999.

## 1.2 Goals

The goals of Mellow Yellow are to specify, develop and test a workable GUI for the CSPN system that will function as designed when interfaced with the port team's code, to produce comprehensive documentation that would enable users with any level of experience to use the system proficiently, and to accomplish these objectives within the specified time frame.

## 1.3 Definitions

- **CSPN** – CSP-to-Stochastic Petri Nets
- **CSP** – Communicating Sequential Processes (process algebra)
- **GUI** – Graphical User Interface
- **Mellow Yellow** – The best software engineering team money can't buy

## 1.4 Organization

This document consists of the following sections:

- **Abstract:** A short description of the purpose, contents and conclusions of the project plan
- **Introduction:** A summary of the goals of this project and definitions of terms used within this document
- **Partitioning Strategy:** A list of team members and their roles in the completion of the project
- **Basic Functional Description:** An overview of the CSPN GUI, with suggested enhancements and a context diagram
- **Development Environment:** A list of the hardware and software tools and standards that Mellow Yellow will use during this project
- **Risk Assessment:** An enumeration of the risks to the project as we perceive them at this point in time

- **Testing Approach:** A description of the testing philosophies we plan to employ
- **Schedule:** A tentative schedule for development and deliverables

## 2. Partitioning Strategy

- **Kerry Hammil:** Group Leader and in charge of project management, writing and website maintenance. Kerry will handle the overall operations of the project, write the formal project in the format necessary, and also keep the group webpage up to date as it will be changing throughout the course of the project.
- **Alex Velkov:** Writing the documentation and specifications of the project. Alex will be in charge of documenting all of the code. He will also do the design specifications of the project as it progresses.
- **Nick Gunder:** Development, FTP site maintenance and version control. Nick will be involved in the coding of the project and also keep the FTP site up to date as needed. He will also be in control of version control which is to keep track of the enhancements that are made after each testing phase of the project.
- **Ryoji Noda:** Development. Ryoji will be one of the main group members to deal with coding and implementation portion of the project.
- **Azrina Hussin:** Azrina will participate in the testing of the code after each phase of implementation, ensuring that the amount of bugs in the code is minimal and that the code functions as specified.
- **Brock Rogers:** Brock will work with Azrina on testing.

### 3. Basic Functional Description

The CSPN Graphical User Interface (GUI<sup>1</sup>) functionally consists as three main sections:

1. The Command Line Communication and Interpretation Section.
2. Third Party Application and Format Support Section.
3. The Interaction Section.

Features listed below in italics are proposed enhancements to the project.

#### 3.1 *The Command Line Communication and Interpretation Section*

The GUI will provide the user with the same capabilities as the command line switches do in the command-line version of the utility. The GUI will send then send the user's selections to the port.

#### 3.2 *Third Party Application and Format Support Section*

Some of the added functionality will need to support third part applications and formats. Here are the features that will be standard in the CSPN GUI:

- Third party application launcher support; this will allow the user to load his or her favorite editor to create and modify input files.
- *The GUI shall support the loading and display of the CompuServe Graphics Interchange (GIF) graphic file format.*
- *The GUI shall support the loading and display of the Windows or OS/2 (Bitmap) graphic file format.*
- *The GUI shall support the loading and display of the JPEG Compliant graphic file format.*
- The GUI shall include the support for loading postscript text files.

#### 3.3 *The Interaction Section*

Interaction with the user will account for more then 60% of the total development effort in the CSPN GUI.

These are the features that will allow the user to interact with the GUI:

- Context Sensitive Help.
- Language Constructs (available in the CSPN User Manual).

---

<sup>1</sup> Pronounced Goo-Wee.

- A Rudimentary Editor for use in creating and modifying input files.
- *A non-standard widget library*<sup>2</sup>.
- *Printing of input files.*
- *Sounds.*
- *Macros.*

---

<sup>2</sup> Trying to move away from the “Microsoft” type User Interface design.

---

# Context Diagram

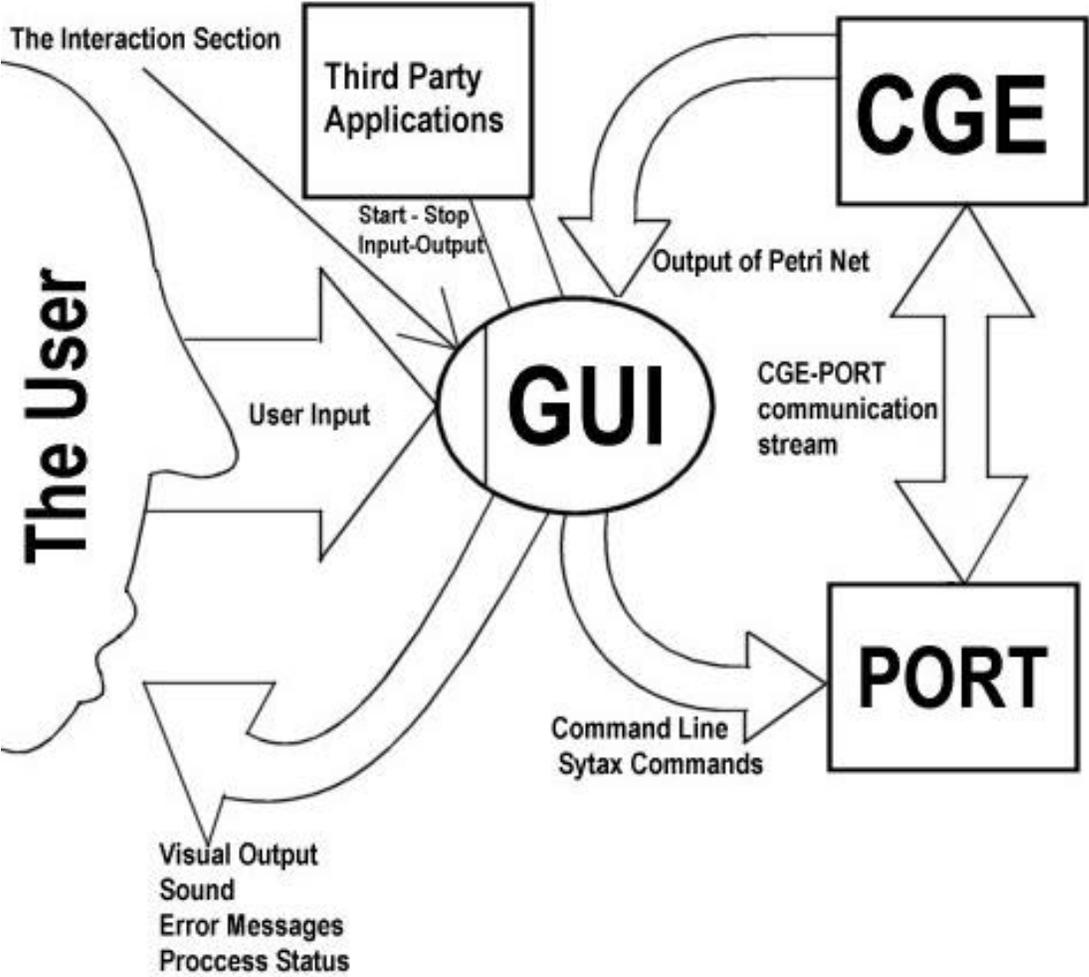


Figure 1: CSPN GUI Context Diagram

---

## **4. Development Environment**

### *4.1 Software*

We will use Microsoft Visual Studio 6.0 and the MSDN Library for our project.

### *4.2 Platform*

Our project will run under Windows 95/98/NT

### *4.3 Libraries*

We have the Windows API and the Microsoft Foundation Classes (MFC) available to us for development

## 5. Risk Assessment

- The primary risk at this point of the software engineering process is that we do not have a very good idea of what the product does. Obviously, approaching a project with limited knowledge is a sure recipe for disaster. We plan to remedy this by meeting with the customer on 25 Sep 99 at which we hope to hammer out the details that we will need to create a successful product.
- Another risk is that we have not decided what language we will be using to design the product. Our two choices are C++ and Java. The two developers seem to prefer C++, so that will probably be the choice we will make. However, the wrong choice of language will make our project that much more difficult, so this decision will have to be weighed carefully.
- Inter-group communication is a risk as well. The group might not be able to get a hold of a member prior to an important meeting or deadline, resulting in lost productivity. To counter this risk, we will try to contact members several days before they are needed, using email as well as a telephone. We have also discussed a policy on missing group meetings or deadlines.
- A short-term risk to the project is that Brock is ill this week and unable to come to meetings. However, we do not anticipate a problem with that, since he receives progress reports of every meeting.
- An unlikely risk to the project is a group member dropping the class. This would put the team in a bind and increase the workload for everyone. However, every team member seems dedicated to the class, and this situation should not arise.
- Since our code is intended to interface with that generated by the port team, that interface becomes a risk to us. We need to negotiate a common standard so that our project can communicate with theirs, and we are dependent on them to deliver their code soon enough that we can test ours against it.

## 6. Testing Approach

Software testing is an element that is often referred as *verification and validation (V&V)*.

*Verification* answers the question “Are we building the product right?”, and *validation* answers the question “Are we building the right product?”.

### 6.1 General Strategies

Generally, these are the testing strategies that we plan to use for the software project:

- *Bottom-up testing*

This testing starts with the fundamental components and proceeds to the higher level. We plan to use this testing because any design errors can be detected at an early stage of the testing process. We can avoid extensive re-design and re-implementation of the software thus save costs and time.

- *Stress testing*

This testing relies on stressing the software by going beyond its limits. This will test how well the system cope with overload situation. We will specify the maximum load for the software and then test it until the maximum load is achieved. If everything goes fine, we will add more loads to the system, see how it behaves toward the situation, and fix and continuously test it until the system act correctly.

### 6.2 Defect Testing

This is an approach to discover defects in the software. Three approaches that can be used are black-box testing, white-box testing and interface testing.

- Black-box testing also known as functional testing is the set of tests that are derived from the specification of the system. The ‘black-box’, that is the system can be studied from its inputs and outputs. We will test from all of the aspects of the system and study the outputs.
- White-box testing or structural testing will need us to study the code and the structure of the system to derive the test inputs. This way, we will know how much test data are needed in order to cover the testing requirement for the system. This test will need us to test all path in the system, whether for true or false conditions so that we can see the correctness of the system.

- Interface testing is used to detect the errors when the system combined with the interfaces, whether the errors come from the interfaces or invalid assumptions about the interfaces.

There will be also some tests for specialized environment like GUI. The guidelines for testing GUI include the tests for data entry, windows, mouse operations, keyboard equivalents, menus, and buttons.

We can't explain our testing approaches in more detail at this point because we don't know enough about the project requirements. As we define the requirements and specifications for our product, our test approaches will evolve and we will begin to develop test cases.

## 7. Development and Deliverables Schedule

This schedule is presented as a list of dates and events. Where indicated, specific group members are responsible for the deliverable or milestone. We will further subdivide these events and assign individual portions to team members as the requirements for these deliverables become clearer.

Table 1: Development and Deliverables Schedule for CSPN GUI Project, Team Mellow Yellow

<i>Start Date</i>	<i>End Date</i>	<i>Deliverable/Milestone</i>	<i>Responsible Team Members</i>
<b>9/6</b>	<b>9/10</b>	Project Plan	
<b>9/10</b>	<b>9/17</b>	First Draft of SRS (Software Requirements Specification)	
<b>9/10</b>	<b>9/17</b>	Set up team Web site	Kerry Hammil
<b>9/10</b>	<b>9/17</b>	Set up team FTP site	Nick Gunder
<b>9/10</b>	<b>9/17</b>	Set up development environment on home computers	
<b>9/17</b>	<b>10/11</b>	Prototype of CSPN GUI	Nick Gunder and Ryoji Noda
<b>9/17</b>	<b>10/11</b>	Final Draft SRS	
<b>10/13</b>		PDR (Preliminary Design Review)	
<b>10/25</b>		CDR (Critical Design Review)	
<b>11/1</b>		Design Notebook Due	
<b>11/29</b>		Demonstration and Test Report Due	
<b>12/6</b>		User Manual Due	