Characteristics

Meaningful Data Display
   All data readily accessible
   Tables, Charts, Graphs

Modifiability
   What-you-see-is-what-you-get
   Random access - data can be changed in any order

Constraint Checking
   Completeness and Consistency
   Problem alerting and Guidance in solving

Automatic Computation of Results
   Consequences computed
   Presentation automatically updated
# DEPARTMENT OF COMPUTER SCIENCE
MSCS Program Sheet (2010-11)

<table>
<thead>
<tr>
<th>Name: Charles Pamell Naut</th>
<th>Advisor:</th>
<th>Proposed date for degree conferral:</th>
<th>Date: 10/8/2010</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student ID #:</td>
<td>Email:  <a href="mailto:cnau@stanford.edu">cnau@stanford.edu</a></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

## General Instructions
Before the end of your first quarter, you should complete the following steps. Detailed instructions are included in the Guide to the MSCS Program Sheet in your orientation packet (an online version is available at es.stanford.edu/degrees/mscs/programsheets/):

- Complete this program sheet by filling in the number, name and units of each course you intend to use for your degree.
- Create a course schedule showing the year and quarter in which you intend to take each course in your program sheet.
- Meet with your adviser and secure the necessary signatures on the program sheet.

## Foundations Requirement
You must satisfy the requirements listed in each of the following areas; all courses taken elsewhere must be approved by your adviser on a foundation course waiver form. Required documents for waiving a course include course descriptions, syllabi, and textbook lists. These documents can be organized here: es.stanford.edu/degrees/mscs/waivers/. Do not enter anything in the “Units” column for courses taken elsewhere.

Note: If you are amending an old program sheet, enter "on file" in the approval column for courses that have already been approved.

<table>
<thead>
<tr>
<th>Requirement</th>
<th>Equivalent elsewhere (course number/title/institution)</th>
<th>Approval</th>
<th>Grade</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic, Automata and Complexity (CS 103)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Probability (CS 109, STATS 116, CME 106, or MS&amp;E 220)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Algorithmic Analysis (CS 161)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer Organization and Systems (CS 107)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Principles of Computer Systems (CS 110)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

TOTAL UNITS USED TO SATISFY FOUNDATIONS REQUIREMENT: 10

Note: This total may not exceed 10 units.
Current Approach

MySQL
PHP
JavaScript
CSS
HTML

The Big 5
Do you master them all?
Do It Yourself!

Worksheets :: Spreadsheets
<center>
<input id='mynode'
    type='text'
    value='hello'
    size='30'
    style='color:black'/>
</center>
Predicates

\texttt{value}(\texttt{widget}, \texttt{value}) - true whenever the value associated with \texttt{widget} is \texttt{value}. The widget here may be a text field, selector, radio button field, slider, and so forth.

\texttt{holds}(\texttt{widget}, \texttt{value}) - true whenever one of the values associated with the multi-valued node \texttt{widget}. The widget in this case is a multi-valued selector or a checkbox field.

\texttt{attribute}(\texttt{widget}, \texttt{property}, \texttt{value}) - true whenever the \texttt{property} attribute of \texttt{widget} is \texttt{value}.

\texttt{style}(\texttt{widget}, \texttt{property}, \texttt{value}) - true whenever the \texttt{property} style of \texttt{widget} is \texttt{value}.

\texttt{innerhtml}(\texttt{widget}, \texttt{content}) - true whenever the \texttt{innerHTML} of \texttt{widget} is \texttt{content}. The \texttt{content} is typically a string of characters.
Dataset Representation

**DOM:**

```html
<center>
  <input id='mynode'
          type='text'
          value='hello'
          size='30'
          style='color:black'/>
</center>
```

**Dataset:**

- `value(mynode,hello)`
- `attribute(mynode,size,30)`
- `style(mynode,color,black)`
DOM:

```html
<center>
<input id='mynode'
     type='text'
     value='hello'
     size='30'
     style='color: black'/>
</center>
```

Dataset:

```
value(mynode, hello)
attribute(mynode, size, 30)
style(mynode, color, black)
```
Gestures
Actions

Gestures performed by the user:
Making a selection from drop-down list
Changing value of text field
Clicking a button

Automatic Actions:
Loading a page
Clock tick
**Operations**

`select(widget, value)`: This action occurs when the user enters or selects `value` as the value of `widget`.

`deselect(widget, value)`: This action occurs when the user erases or deselects `value` as the value of `widget`.

`click(widget)`: This action occurs when the user clicks on `widget`.

`tick`: This action occurs periodically. By default, it happens once per second.

`load`: This occurs when a page is first loaded.

`unload`: This action occurs when a user leaves a page.
**Example**

**DOM:**
```html
  <center>
    <input id='mynode'
           type='text'
           value='hello' -> goodbye
           size='30'
           style='color:black'
           onchange='modtext(this)'/>
  </center>
```

**Resulting Action:**
```javascript
  select(mynode, goodbye)
```
Operation Definitions
Buttons

click(orange) :: style(page,color,orange)
click(blue) :: style(page,color,blue)
click(purple) :: style(page,color,purple)
click(black) :: style(page,color,black)

click(X) :: style(page,color,X)

click(X) ::
    style(page,color,Y) & distinct(X,Y)
=> ~style(page,color,Y)
Selector

```
select(pagecolor,X) :: style(page,color,X)
select(pagecolor,X) ::
    style(page,color,Y) ==> ~style(page,color,Y)

select(pagecolor,X) :: value(pagecolor,X)
select(pagecolor,X) ::
    value(pagecolor,Y) ==> ~value(pagecolor,Y)
```
Interaction

click(X) :: value(pagecolor,X)
click(X) ::
    value(pagecolor,Y) ==> ~value(pagecolor,Y)
View Definitions
Alternative Approaches

Direct Approach:
Every gesture directly changes visible features.

View Definitions:
Every gesture changes dataset.
Visible features defined as views of the dataset and are changed implicitly in Worksheets code.
Changing color:

\[
\text{click}(X) :: \text{style}(\text{page}, \text{color}, X) \\
\text{select}(\text{pagecolor}, X) :: \text{style}(\text{page}, \text{color}, X) \\
\text{select}(\text{pagecolor}, X) :: \\
\quad \text{style}(\text{page}, \text{color}, Y) \implies \neg \text{style}(\text{page}, \text{color}, Y)
\]

Changing selector value:

\[
\text{click}(X) :: \text{value}(\text{pagecolor}, X) \\
\text{click}(X) :: \\
\quad \text{value}(\text{pagecolor}, Y) \implies \neg \text{value}(\text{pagecolor}, Y) \\
\text{select}(\text{pagecolor}, X) :: \text{value}(\text{pagecolor}, X) \\
\text{select}(\text{pagecolor}, X) :: \\
\quad \text{value}(\text{pagecolor}, Y) \implies \neg \text{value}(\text{pagecolor}, Y)
\]
Changing selector value:

\[
\text{click}(X) :: \text{value}(\text{pagecolor}, X) \\
\text{click}(X) :: \\
\quad \text{value}(\text{pagecolor}, Y) \implies \neg \text{value}(\text{pagecolor}, Y)
\]

\[
\text{select}(\text{pagecolor}, X) :: \text{value}(\text{pagecolor}, X) \\
\text{select}(\text{pagecolor}, X) :: \\
\quad \text{value}(\text{pagecolor}, Y) \implies \neg \text{value}(\text{pagecolor}, Y)
\]

View Definition:

\[
\text{style}(\text{page}, \text{color}, X) :- \text{value}(\text{pagecolor}, X)
\]
Putting It All Together
Converting Web Pages to Worksheets

Start with an HTML page.

(1) Add worksheets code.
(2) Initialize.
(3) Add identifiers and event handlers.
(4) Add Data and Rules.

Done.
<html>
<head>
</head>
<body>
    <input type='button' value='orange'/>
    <input type='button' value='purple'/>
    <input type='button' value='black'/>
    <br/>Some text.<br/>
    <select>
        <option>orange</option>
        <option>purple</option>
        <option>black</option>
    </select>
</body>
</html>
<html>
  <head>
    <script type='text/javascript'
      src='http://epilog.stanford.edu/javascript/epilog.js'/>
    <script type='text/javascript'
      src='http://worksheets.stanford.edu/javascript/worksheets.js'/>
  </head>
  <body>
    <input type='button' value='orange'/>
    <input type='button' value='purple'/>
    <input type='button' value='black'/>
    <br/>Some text.<br/>
    <select>
      <option>orange</option>
      <option>purple</option>
      <option>black</option>
    </select>
  </body>
</html>
<html>
    <head>
        <script type='text/javascript'
            src='http://epilog.stanford.edu/javascript/epilog.js'/>
        <script type='text/javascript'
            src='http://worksheets.stanford.edu/javascript/worksheets.js'/>
    </head>
    <body onload='initialize()'>
        <input type='button' value='orange'/>
        <input type='button' value='purple'/>
        <input type='button' value='black'/>
        <br/>Some text.<br/>
        <select>
            <option>orange</option>
            <option>purple</option>
            <option>black</option>
        </select>
    </body>
</html>
<html>
<head>
    <script type='text/javascript' src='http://epilog.stanford.edu/javascript/epilog.js'/>
    <script type='text/javascript' src='http://worksheets.stanford.edu/javascript/worksheets.js'/>
</head>
<body id='page' onload='initialize()'>
    <input type='button' value='orange' id='orange' onclick='modbutton(this)'/>
    <input type='button' value='purple' id='purple' onclick='modbutton(this)'/>
    <input type='button' value='black' id='black' onclick='modbutton(this)'/>
    <br/>Some text.<br/>
    <select id='pagecolor' onchange='modselector(this)'>
        <option>orange</option>
        <option>purple</option>
        <option>black</option>
    </select>
</body>
</html>
<html>
  <head>
    <script type='text/javascript'
      src='http://epilog.stanford.edu/javascript/epilog.js'/>
    <script type='text/javascript'
      src='http://worksheets.stanford.edu/javascript/worksheets.js'/>
  </head>
  <body id='page' onload='initialize()'>
    <input type='button' value='orange' id='orange' onclick='modbutton(this)'/>
    <input type='button' value='purple' id='purple' onclick='modbutton(this)'/>
    <input type='button' value='black' id='black' onclick='modbutton(this)'/>
    <br/>Some text.<br/>
    <select id='pagecolor' onchange='modselector(this)'>
      <option>orange</option>
      <option>purple</option>
      <option>black</option>
    </select>
    <textarea id='lambda' style='display:none'></textarea>
    <textarea id='library' style='display:none'>...</textarea>
  </body>
</html>
Some text
Semantic Worksheets
**DEPARTMENT OF COMPUTER SCIENCE**

**MSCS Program Sheet (2010-11)**

Name: Charles Parnell Naut  
Advisor:  
Student ID #:  
Email: chnaau@stanford.edu  
Proposed date for degree conferral:  
Date: 10/8/2010  
☐ HCP?  
☐ Coterm?

### General Instructions

Before the end of your first quarter, you should complete the following steps. Detailed instructions are included in the Guide to the MSCS Program Sheet in your orientation packet (an online version is available at cs.stanford.edu/degrees/mscs/programsheets/):

- Complete this program sheet by filling in the number, name and units of each course you intend to use for your degree.
- Create a course schedule showing the year and quarter in which you intend to take each course in your program sheet.
- Meet with your advisor and secure the necessary signatures on the program sheet.

### Foundations Requirement

You must satisfy the requirements listed in each of the following areas; all courses taken elsewhere must be approved by your adviser on a foundation course waiver form. Required documents for waiving a course include course descriptions, syllabi, and textbook lists. These documents can be organized here: cs.stanford.edu/degrees/mscs/waivers/. Do not enter anything in the “Units” column for courses taken elsewhere.

Note: If you are amending an old program sheet, enter "on file" in the approval column for courses that have already been approved.

<table>
<thead>
<tr>
<th>Required</th>
<th>Equivalent elsewhere (course number/title/institution)</th>
<th>Approval</th>
<th>Grade</th>
<th>Units</th>
</tr>
</thead>
<tbody>
<tr>
<td>Logic, Automata and Complexity (CS 103)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Probability (CS 109, STATS 116, CME 106, or MS&amp;E 220)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Algorithmic Analysis (CS 161)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Computer Organization and Systems (CS 107)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Principles of Computer Systems (CS 110)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Total Units Used to Satisfy Foundations Requirement:** 10

Note: This total may not exceed 10 units.
Heterogeneous Worksheets
Collaborative Heterogeneous Worksheets
Architectural Alternatives
Syntactic vs Semantic Worksheets

**Syntactic Worksheets**

*Widget state* (e.g. value of selector) stored
User gestures (e.g. clicking a button) change *widget state*
Visible Features (e.g. color of text) views of *widget state*

**Semantic Worksheets**

*Application state* (e.g. courses student has taken) stored
User gestures translated to *application actions*
Visible Features computed as *views of application state*
<table>
<thead>
<tr>
<th>Course 1</th>
<th>Course 2</th>
<th>Course 3</th>
<th>Course 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Autumn</td>
<td>Autumn</td>
<td>Autumn</td>
<td>Autumn</td>
</tr>
<tr>
<td>Winter</td>
<td>Winter</td>
<td>Winter</td>
<td>Winter</td>
</tr>
<tr>
<td>Spring</td>
<td>Spring</td>
<td>Spring</td>
<td>Spring</td>
</tr>
<tr>
<td>Summer</td>
<td>Summer</td>
<td>Summer</td>
<td>Summer</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Autumn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course 1</td>
</tr>
<tr>
<td>Course 2</td>
</tr>
<tr>
<td>Course 3</td>
</tr>
<tr>
<td>Course 4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Winter</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course 1</td>
</tr>
<tr>
<td>Course 2</td>
</tr>
<tr>
<td>Course 3</td>
</tr>
<tr>
<td>Course 4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Spring</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course 1</td>
</tr>
<tr>
<td>Course 2</td>
</tr>
<tr>
<td>Course 3</td>
</tr>
<tr>
<td>Course 4</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Summer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course 1</td>
</tr>
<tr>
<td>Course 2</td>
</tr>
<tr>
<td>Course 3</td>
</tr>
<tr>
<td>Course 4</td>
</tr>
</tbody>
</table>
Operation Definitions:

select(Course, Quarter) :: holds(Course, Quarter)
deselect(Course, Quarter) :: ~holds(Course, Quarter)

Widget Data:

holds(course1, autumn)
holds(course2, autumn)
Schedule 2

Operation Definitions:

\[
\text{select}(\text{Quarter}, \text{Course}) :: \text{holds}(\text{Quarter}, \text{Course}) \\
\text{deselect}(\text{Quarter}, \text{Course}) :: \neg\text{holds}(\text{Quarter}, \text{Course})
\]

Widget Data:

\[
\text{holds(autumn,course1)} \\
\text{holds(autumn,course2)}
\]
Syntactic Mapping Rules

Mapping Rules for Schedule 1 to Schedule 2:

select(Quarter,Course) :: holds(Course,Quarter)
deselect(Quarter,Course) :: ~holds(Course,Quarter)

Mapping Rules for Schedule 2 to Schedule 1:

select(Course,Quarter) :: holds(Quarter,Course)
deselect(Course,Quarter) :: ~holds(Quarter,Course)

Widget Data:

holds(course1,autumn)
holds(course2,autumn)
holds(autumn,course1)
holds(autumn,course2)
Data:
offered(course1,autumn)
offered(course2,autumn)

Operations:
select(Course,Quarter) :: offered(Course,Quarter)
deselect(Course,Quarter) :: ~offered(Course,Quarter)

select(Quarter,Course) :: offered(Course,Quarter)
deselect(Quarter,Course) :: ~offered(Course,Quarter)

Views:
holds(Course,Quarter) :- offered(Course,Quarter)
holds(Quarter,Course) :- offered(Course,Quarter)
### AI Program Requirements

- **Take at least 6 courses**
- **Take at most 3 courses per quarter**

<table>
<thead>
<tr>
<th>Course</th>
<th>Units</th>
<th>Total</th>
<th>Professor Courses</th>
</tr>
</thead>
<tbody>
<tr>
<td>CS 124</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 131</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 157</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CS 223A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 224N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 225A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 227B</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CS 228</td>
<td></td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>CS 229</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 231N</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 238</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 273A</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 273B</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 276</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 279</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>CS 331B</td>
<td></td>
<td>3</td>
<td></td>
</tr>
</tbody>
</table>

#### Focus on AI Topics

- **25%** Computational Biology
- **25%** Computer Vision
- **25%** Information Retrieval
- **25%** Logic
- **Machine Learning**
- **Natural Language Processing**
- **Robotics**

**Quarters:**
- **Autumn**
- **Winter**
- **Spring**

**Total:** 12 units
Collaborative Worksheets
Nineboard
Collaborative Nineboard
Tic Tac Toe

X O X
Trifecta

Deck
- 1,2
- 1,3
- 2,1
- 2,2
- 2,3
- 3,3

White
- 1,1
- 3,1

Black
- 3,2
Tic Tac Toe - Trifecta
Architectural Choices

Dataset Sharing
Easy to implement and debug
May move lots of data
Allows all users to see and modify all data

Message Passing (Communication Channels)
Difficult to implement and debug
Moves minimal data
Privacy and security assured

Backend Server (MySQL, PHP, etc.)
Moderate effort to implement and debug
Development and maintenance of backend infrastructure
Moves minimal data
Privacy and security assured
Worksheets

Create dynamic, interactive web pages. Publish online for personal or public use. Interlink to support collaborative work.

Public
Click here to access public worksheets.

Private
Click here to manage your own worksheets.

Cloud-based. No need to install hardware or software. Do It Yourself. No traditional programming required. Easy to manage. Dashboards and drop down lists.

Learn More