**Abstract:**
BGraph is a compiler from a language, which is created specifically to define data graphs, to JGraph code, which is another language to draw graphs or arbitrary shapes. Later on JGraph can be used to generate PostScript output.

**Old Solution - JGraph:**
- JGraph is a compiler from JGraph code to PostScript.
- It is also a language designed to generate graphs.

**Advantages of JGraph are:**
- Output is very customizable, since language has features to draw lines, curves and text at any arbitrary point.
- Input can be read from shell commands to import any kind of data.

**Disadvantages are:**
- The language is very low level and generates a lot of complexity.
- Shell commands work like macros. Every shell commands is first executed and that part of the code is replaced with the standard output from the command. Only the final code without any shell commands is compiled.
- Positioning system is relative to axis origin and range of the axis. Therefore every position is data dependent including titles, arbitrary texts.
- No natural way to generate some kind of graphs, for example where bars are stacked.

**New Solution - BGraph:**
- BGraph language is specially designed to naturally represent bar graphs where bars can be grouped, stacked or both. It can be seen as a wrapper for JGraph.

**Advantages of BGraph are:**
- Graph definition can be totally independent from data since data can be read from any external source plug-in framework.
- Output is customizable enough to represent most graphs, but customizations are optional and very nicely integrated with the language to allow for a representation that is human readable.
- Code of BGraph has been designed keeping extensibility and maintainability in mind from the beginning. Therefore new features, output formats, plug-ins can be added with minimum work.

**Disadvantages of BGraph are:**
- Only bar graphs are supported in BGraph and currently there is no other way to draw other type of graphs.
- Current implementation, which outputs JGraph, is limited by JGraph’s positioning system but tries to place items as close as possible to their specified locations.

**Code Structure:**
- BGraph is coded in Java 1.4 for compatibility.
- JavaCC is used to specify language grammar and generate the parser code.
- Code is designed carefully to be easily maintained and extended.

**Input File Structure:**
- Includes Templates from other files can be loaded using includes.

**Array Definitions and Style Templates**

**Graph Definition**
- Size – Defines the size of drawing space
- Title properties – Optional, used to set title text and properties
- Y-Axis and X-Axis properties
- Legend – If a legend is wanted, must be used to set its properties

**Data Value Entries**
- Defines the sparse matrix to store data values, example: data\{group\}[\{bar\}]\{\{0;5\}\{0\} = \{0\}; \{0\}; \{10, 250, 255\};

**Chart Style Entries**
- Used to specify style of each bar or stack in the graph, example: style\{\{0\}\{0\} = \{fill-color = \{10, 250, 255\};\}

**Label Entries**
- Used to specify label for each bar, stack, group of bars. These labels form the legend or labels in the x-axis, example:

```
labels \{\{0;1\}\{0\} = \{plugin:FileRead2D {
  filename = \$files\[\$i\]; line = \{ \$i + 3 \}; column = 1;
} };
```

**Data Input:**
- Data in BGraph is represented as a 3-dimensional matrix. User can choose what each dimension represents and also choose to ignore unused dimensions.
- Data types are very flexible. BGraph supports numbers, strings, arrays of any values and most importantly plug-ins.

**Example:**
```
data\{group\\{bar\}\} \{\{0;1\}\{0;3\} = \{plugin:FileRead2D {
  filename = \"input.txt\";
  line\[\$i\]; column\[\$j\];
} }
```

This code generated a matrix of grouped bars where values are read from a file containing a 2D table by iterating through the specified ranges.

**Conclusion:**
BGraph is developed while trying to keep a balance between customizability and ease of use. It can currently be used to generate a range of data graphs using bars, but it is also open to extensions to support a wider variety of graphs and features.