**Software Construction**

**CRC Cards with Contracts**

**for Control Flow Graph Generator**

**06/27/2016**

**Team 7**

**Figure 1:** GUI CRC Card

|  |  |
| --- | --- |
| **Class Name:** GUI | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:** Class that displays CFG with graphical elements. | |
| **Contracts:**  **1. Capture Input/Display Output**   * Retrieve user filename input. * Send filename. * Display receipt notification.   **Private Responsibilities:**  **Visualize Graph**   * Retrieve XML file. * Display contents (by means of external library or other). | **Collaborations:**    XML(1) |
| **Comments:**  Class in charge of visualizing the created CFG. The visualization library might be a web service as D3, Neo4j, or similar. | |

**Figure 2:** XML CRC Card

|  |  |
| --- | --- |
| **Class Name:** XML | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:**This class is in charge of providing an XML service with instructions represented with tags. | |
| **Contracts:**  **1. Create XML File**   * Generate XML file. * Return XML file.   **Private Responsibilities:**  **Create Node XML**   * Retrieve nodes. * Generate XML tags.   **Create Edge XML**   * Retrieve edges. * Generate XML tags. | **Collaborations:**  CFG(1)  CFG(1) |
| **Comments:** | |

**Figure 3:** Control Flow Graph CRC

|  |  |
| --- | --- |
| **Class Name:** Control Flow Graph | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:** It is in charge of building a control flow graph by the use of the parsed meaning of each statement. Building a CFG is a composition of the following components: node, edge, exit/entry points, and parsed statement instruction. | |
| **Contracts:**    **1. Get CFG Data.**   * Returns data structure of nodes, edges, and entry and exit points.   **Private Responsibilities:**  **Build Node(s)**   * Retrieve type of parsed statement. * Create node with statement.   **Build Edge(s)**   * Retrieve type of parsed statement. * Create edge between node.     Knows type of parsed statements. | **Collaborations:**        Parser(1)  Node(1)  Parser(1)  Edge(1) |
| **Comments:** Class that is the intermediary between parsed statements from the file and the XML file with its tags. | |

**Figure 4:** Edge CRC Card

|  |  |
| --- | --- |
| **Class Name:** Edge | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:** A directed path from one node to another. | |
| **Contracts:**  **1. Build Edge**   * Create edge. * Return edge.   **Private Responsibilities:**  Has “from” node.  Has “to” node. | **Collaborations:** |
| **Comments:**  Entry point and exit point for each node. | |

**Figure 5:** Node CRC Card

|  |  |
| --- | --- |
| **Class Name:** Node | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:** It is the representation of a basic block. | |
| **Contracts:**  **1. Build Node**   * Create node. * Return node.   **Private Responsibilities:**  Has unique identifier.  Has “coming from” edges.  Has “going to” edges. | **Collaborations:** |
| **Comments:**  Class that contains corresponding lines reflecting the sentence, node identifier | |

**Figure 6:** Parser CRC Card

|  |  |
| --- | --- |
| **Class Name:** Parser | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:** Uses a set of rules provided by a grammar to parse statements. | |
| **Contracts:**  **1. Send Parsed Data**   * Retrieve parsed data. * Send parsed data.   **Private Responsibilities:**  **Parse File**   * Retrieve statements from file. * Call rules for each statements. * Parse statements. * Store parsed data.   **Provide Statement Type**   * Retrieve statement type rules. * Determine statement type. * Get and return statement of type. | **Collaborations:**    Grammar(1)  Statement(1) |
| **Comments:** | |

**Figure 7:** Grammar CRC Card

|  |  |
| --- | --- |
| **Class Name:** Grammar | |
| **Superclass:** | |
| **Subclasses:** | |
| **Description:** A set of syntax rules for a given language. | |
| **Contracts:**  **1. Provide Grammar Rules**   * Retrieve rules from file. * Return rules.   **Private Responsibilities:**  Knows terminals.  Knows nonterminals. | **Collaborations:** |
| **Comments:** | |

**Figure 8:** Statement CRC Card

|  |  |
| --- | --- |
| **Class Name:** Statement | |
| **Superclass:** | |
| **Subclasses:** Sequential, Conditional, Iterative | |
| **Description:** Syntax that defines the statements. | |
| **Contracts:**  **1. Create Statement**   * Retrieve statement’s type. * Create statement of type.   **Private Responsibilities:**  Identify terminals.  Identify non-terminals.  Identify basic blocks. | **Collaborations:** |
| **Comments:**  Parent class that determines type of statements that is being applied.  A statement is composed by terminal, non terminal. Determine statement type is based on the terminal,non-terminal | |

**Figure 9:** Conditional CRC Card

|  |  |
| --- | --- |
| **Class Name:** Conditional | |
| **Superclass:** Statement | |
| **Subclasses:** | |
| **Description:** Syntax rules for conditional statements. | |
| **Contracts:**  **Private Responsibilities:**  Identifies conditional statements.  Send conditional block. | **Collaborations:** |
| **Comments:** | |

**Figure 10:** Sequential CRC Card

|  |  |
| --- | --- |
| **Class Name:** Sequential | |
| **Superclass:** Statement | |
| **Subclasses:** | |
| **Description:** Syntax rules for sequential statements. | |
| **Contracts:**  **Private Responsibilities:**  Identifies sequential statements.  Send sequential block. | **Collaborations:** |
| **Comments:** | |

**Figure 11:** Iterative CRC Card

|  |  |
| --- | --- |
| **Class Name:** Iterative | |
| **Superclass:** Statement | |
| **Subclasses:** | |
| **Description:** Syntax rules for iterative statements. | |
| **Contracts:**  **Private Responsibilities:**  Identify iterative statements.  Send iterative block. | **Collaborations:** |
| **Comments:** | |