|  |  |
| --- | --- |
| **Class Name:** Action | |
| **Description:** The Abstract class of operations to perform Actions on a Field with a Formatter’s Rule(s) | |
| **Contracts:**  1. Manage Action  *Create Action*  createAction(Field)  Description: Creates an Action to this Field  Pre:  Post:  *Delete Action*  deleteAction()  Description: Deletes this Action  Pre:  Post:  2. Manage Field Visualization  *Hide Field*  hideField(Field)  Description: Receives a Field to be marked as hidden.  Pre:  Post:    *Show Field*  showField(Field)  Description: Receives a Field to be marked as displayed.  Pre:  Post:  3. Manage Field Annotation  *Add an Annotation to a Field*  addAnnotation(Field, String)  Description: Receives a Field and a String to be linked as an Annotation to this Field.  Pre:  Post:  *Delete the Annotation to a Field*  addAnnotation(Field, String)  Description: Receives a Field and a String to be linked as an Annotation to this Field.  Pre:  Post:  4. Manage Field Hook  *Add a Hook to a Field*  addHook (Field, Hook)  Description: Receives a Field and a Hook to be linked to this Field.  Pre:  Post:  *Delete Hook from Field*  deleteHook(Field, Hook)  Description: Receives a Field and a Hook to be linked to this Field.  Pre:  Post:  5. Manage Field Renaming  *Rename Field*  renameField(Field, String)  Description: Receives a Field and a String to rename this Field.  Pre:  Post:  **Private Responsibilities:** | **Collaborations:**  Rule (23), Hook (13)  Rule (23), Hook (13)  Rule (23), Hook (13)  Field (5)  Field (5)  Field (5)  Field (6), Hook (15)  Field (6), Hook (15)  Field (5), GUI (28) |
| **Comments:**  Contains the steps by which to: Hide, Rename, Annotate, Create Hook for a Field | |

|  |  |
| --- | --- |
| **Class Name:** HidingAction | |
| **Description:** Contains the operations to perform Hiding a Field with a Formatter’s Rule(s) | |
| **Contracts:**  1. Manage Action  *Create Action*  createAction()  Description: Creates an Action  Pre:  Post:  *Delete Action*  deleteAction()  Description: Deletes this Action  Pre:  Post:  2. Manage Field Visualization  *Hide Field*  hideField(Field)  Description: Receives a Field to be marked as hidden.  Pre:  Post:    *Show Field*  showField(Field)  Description: Receives a Field to be marked as displayed.  Pre:  Post:  **Private Responsibilities:** | **Collaborations:**  Rule (23)  Rule (23)  Field (5)  Field (5) |
| **Comments:**  Contains the steps by which to: Hide a Field | |

|  |  |
| --- | --- |
| **Class Name:** AnnotationAction | |
| **Description:** Contains the operations to perform Annotations on a Field with a Formatter’s Rule(s) | |
| **Contracts:**  1. Manage Action  *Create Action*  createAction(Field)  Description: Creates an Action to this Field  Pre:  Post:  *Delete Action*  deleteAction()  Description: Deletes this Action  Pre:  Post:  3. Manage Field Annotation  *Add an Annotation to a Field*  addAnnotation(Field, String)  Description: Receives a Field and a String to be linked as an Annotation to this Field.  Pre:  Post:  *Delete the Annotation to a Field*  addAnnotation(Field, String)  Description: Receives a Field and a String to be linked as an Annotation to this Field.  Pre:  Post:  **Private Responsibilities:** | **Collaborations:**  Rule (23)  Rule (23)  Rule (23), Field (5), GUI (28)  Rule (23), Field (5), GUI (28) |
| **Comments:**  Contains the steps by which to: Annotate a Field | |

|  |  |
| --- | --- |
| **Class Name:** HookAction | |
| **Description:** Contains the operations to perform Actions on a Field with a Formatter’s Rule(s) | |
| **Contracts:**  1. Manage Action  *Create Action*  createAction(Field)  Description: Creates an Action to this Field  Pre:  Post:  *Delete Action*  deleteAction()  Description: Deletes this Action  Pre:  Post:  4. Manage Field Hook  *Add a Hook to a Field*  addHook (Field, Hook)  Description: Receives a Field and a Hook to be linked to this Field.  Pre:  Post:  *Delete Hook from Field*  deleteHook(Field, Hook)  Description: Receives a Field and a Hook to be linked to this Field.  Pre:  Post:  **Private Responsibilities:** | **Collaborations:**  Rule (23), Hook (13)  Rule (23), Hook (13)  Rule (23), Field (5), Hook (13)  Rule (23), Field (5), Hook (13) |
| **Comments:**  Contains the steps by which to: Hide, Rename, Annotate, Create Hook for a Field | |

|  |  |
| --- | --- |
| **Class Name:** RenameAction | |
| **Description:** Contains the operations to perform Renaming on a Field with a Formatter’s Rule(s) | |
| **Contracts:**  1. Manage Action  *Create Action*  createAction(Field)  Description: Creates an Action to this Field  Pre:  Post:  *Delete Action*  deleteAction()  Description: Deletes this Action  Pre:  Post:  5. Manage Field Renaming  *Rename Field*  renameField(Field, String)  Description: Receives a Field and a String to rename this Field.  Pre:  Post:  *Restore Old Name*  **Private Responsibilities:** | **Collaborations:**  Rule (23)  Rule (23)  Rule (23), Field (5) |
| **Comments:**  Contains the steps by which to: Rename a Field | |

|  |  |
| --- | --- |
| **Class Name:** Expression | |
| **Description:** The string by which to Filter views on the Proto Element, Field, or Rules | |
| **Contracts:**  6. Manage Primitives  *Add Primitive*  addPrimitive(String)  Description: Receives a primitive (composed from the string) and adds it to this Expression  Pre: The primitive is in the Berkeley Packet Filter (BPF) syntax.  Post: The addition of a primitive to this Expression.  *Remove Primitive*  deletePrimitive(String)  Description: Deletes a specified primitive.  Pre:  Post: The deletion of the primitive from this Expression.  **Private Responsibilities:** | **Collaborations:**  GUI (28)  GUI (28) |
| **Comments:**  Contains one or more primitives defined by Berkeley Packet Filter (BPF) syntax | |

|  |  |
| --- | --- |
| **Class Name:** File | |
| **Description:** Represents the Script, Historical Copy, and PDML | |
| **Contracts:**  7. Create Historical Copy  *Convert in tShark*  convertPCAP(PCAP)  Description: Receives a PCAP file to be sent to tShark for conversion to PDML to be named by this String.  Pre: The file is of type PCAP.  Post:  *Create PDML File*  createHistoricalCopy(PDML)  Description: Receives a PDML file and creates a historical copy of the PDML file.  Pre: The file is a PDML file.  Post:  **Private Responsibilities:**  - Create new File | **Collaborations:**  tShark (27) |
| **Comments:**  Abstract class which handles creation, opening, saving, and modifications to PDML original and Script as well as creation of Historical Copies | |

|  |  |
| --- | --- |
| **Class Name:** Filter | |
| **Description:** Contains Expression(s) by which to visualize and modify the PDML capture | |
| **Contracts:**  8. Manage Filter  *Create Filter*  addFilter(Expression)  Description: Receives an Expression by which to comprise a Filter  Pre:  Post:  *Delete Filter*  deletFilter()  Description: Deletes this Action  Pre:  Post:  9. Manage Expression  *Add to Expression*  addToExpression(String,Expression)  Description: Receives a primitive (composed from the string) and adds it to the Expression  Pre: The Expression is in the Berkeley Packet Filter (BPF) syntax.  Post: The addition of a primitive to the Expression for this Filter.  *Remove Expression*  deleteExpression(Expression)  Description: Deletes a specified Expression.  Pre:  Post:  **Private Responsibilities:** | **Collaborations:**  Rule (22)  Rule (22)  Expression (4)  Expression (4) |
| **Comments:**  Contains one Berkeley Packet Filter (BPF) expression | |

|  |  |
| --- | --- |
| **Class Name:** Formatter | |
| **Description:** Allows to apply rules to Fields in a Protocol for inspection | |
| **Contracts:**  10. Manage Formatter  *Create Formatter*  createFormatter(String)  Description: Builds a Formatter to this new name (keeping to Protocol)  Pre:  Post: Creates a Formatter with an empty list of Rules  *Apply Rules*  applyRules(PDML)  Description: Applies the list of these Rules to the Formatter.  Pre: The list is not empty.  Post:  11. Manage Rules  *Add a New Rule*  addRule(Rule)  Description: Receives a formatter and a rule, adds the rule to the formatter.  Pre: The Rule is not empty and the Formatter exists.  Post:  *Delete the Last Rule*  deleteRule()  Description: Deletes the last rule appended to the Formatter.  Pre:  Post: Other rules remain unmodified, order not affected. | **Collaborations:**  PDML (17)  Rule (22, 23)  Rule (22, 23)  Rule (22, 23) |
| **Comments:**  Contains Filter and Action rules specific to one Protocol; type of File | |

|  |  |
| --- | --- |
| **Class Name:** GUI | |
| **Description:** Interfaces to PFS functionality | |
| **Contracts:**  28. Visualize Script Requests  *Accept Script Modifications*  modifyScript(Hook)  Description: Accepts the user input to modify the Script for this Hook  Pre: This Hook exists  Post:  *Accept Input Parameters*  addInputParameters(Hook, String)  Description: Accepts the user input to add input parameters to run this Hook’s Script  Pre: This Hook exists  Post:  *Accept Link to Script*  modifyScriptLink(Hook, String)  Description: Accepts the user input to modify the link to this Hook’s Script  Pre: This Hook exists and this String is a valid system path  Post:  29. Visualize PDML Modifications  *Revert PDML to Historical Copy*  revertToOriginal(PDML)  Description:  Pre: This PDML has a Historical Copy  Post: The Historical Copy remains unmodified.  *Accept PDML Changes*  acceptChanges(PDML)  Description: Saves edits to this PDML File.  Pre:  Post: The Historical Copy remains unmodified and PDML File is saved.  **Private Responsibilities:**  - Accept input  - Visualize changes | **Collaborations:**  Hook (14)  Hook (14)  Hook (14)  PDML (18)  PDML (18) |
| **Comments:**  Accepts inputs and outputs to display | |

|  |  |
| --- | --- |
| **Class Name:** Historical Copy | |
| **Description:** File Class subtype which contains captured network data within PFS | |
| **Contracts:**  12. Manage Historical Copy  *Create PDML File*  createHistCopy(File)  Description: Receives a PDML file and creates a historical copy of the PDML file.  Pre: The file is a PDML file.  Post:  *Store PDML File*  storePDML(File)  Description: Receives a PDML file and stores it on the local filesystem.  Pre: The file has not been stored before on the current local filesystem of the user.  Post:  **Private Responsibilities:** | **Collaborations:**  File (7)  File (7) |
| **Comments:**  Attributes include identification as unmodifiable and metadata; type of File | |

|  |  |
| --- | --- |
| **Class Name:** Hook | |
| **Description:** Steps to update a value of specific attributes of a Field | |
| **Contracts:**  13. Manage Hook  *Create Hook*  createHook(Script)  Description: Creates a Hook with a Link to a Script  Pre: The Script file exists.  Post:  *Delete Hook*  deletHook()  Description: Deletes this Hook  Pre:  Post:  14. Modify Hook  *Get Input Parameters*  setScriptParameters(Args<>)  Description: Stores the Args as the input parameters for this Hook’s Script  Pre: The Script exists, the number of parameters matches for the Script, and the types match.  Post:  *Modify Link to Script*  modifyLinkToScript(String)  Description: Stores the String as the link or file path to the Python Script  Pre: The link is a valid system path to a Python (.PY) File.  Post:  15. Apply Hook  *Run Script*  runScript()  Description: The Script begins execution  Pre: This Script has its parameters and runs without error  Post:  *Get Return Value to Modify Field*  getReturnValue()  Description: Gets the return value from the Script’s execution  Pre: The Script has completed execution and has a return value  Post:  **Private Responsibilities:** | **Collaborations:**  Action (4), Script (24)  Action (4)  GUI (28)  GUI (28)  GUI (28)  GUI (28) |
| **Comments:**  Contains path to Script, Field to modify, and input parameters to Script’s algorithm (*associated values of specific attributes defined in the Field*) | |

|  |  |
| --- | --- |
| **Class Name:** PDML | |
| **Description:** Allows by subtype to modify and analyze network data within PFS | |
| **Contracts:**  17. Apply Formatter  *Apply Formatter Rules*  applyFormatter(Formatter)  Description: Receives a Formatter to apply its rules to the current PDML  Pre: The Formatter is not currently being edited and the current PDML has Protocol(s) for this Formatter.  Post: The modified PDML will be saved and the Historical Copy will remain unedited.  *Save Changes*  saveChanges()  Description: Stores the PDML on the local system to be saved  Pre:  Post: Current directory is writable by this system.  18. Revert to Historical Copy  *Visualize Changes from PDML and Historical Copy*  findChanges(HistoricalCopy)  Description: Returns a list of the differences between this PDML and its Historical Copy.  Pre: This PDML’s Historical Copy does exist within PFS.  Post: The PDML and Historical Copy will not be changed.  rejectChanges(HistoricalCopy)  Description: Reverts the modified PDML to its Historical Copy  Pre: The Historical Copy exists in the PFS.  Post: The modified PDML copy is a deep copy of its Historical Copy.  **Private Responsibilities:** | **Collaborations:**  Formatter (10), GUI (29)  GUI (29)  Historical Copy (12), GUI (29)  Historical Copy (12), GUI (29) |
| **Comments:**  Attributes include identification as modifiable and metadata; type of File | |

|  |  |
| --- | --- |
| **Class Name:** Protocol | |
| **Description:** Contains Packets by which the matching Formatter can modify | |
| **Contracts:**  19. Visualize Protocol  *Show Protocol*  showProtocol(Protocol)  Description: Receives a protocol currently hidden and displays it.  Pre: The protocol exists within the PDML and is hidden.  Post:  *Hide Protocol*  hideProtocol(Protocol)  Description: Receives a protocol currently displayed and hides it.  Pre: The protocol exists within the PDML and is not hidden.  Post:  **Private Responsibilities:** | **Collaborations:**  Formatter (10)  Formatter (10) |
| **Comments:**  Contains one or more Packets | |

|  |  |
| --- | --- |
| **Class Name:** ProtocolIdentifier | |
| **Description:** Scans PDML to identify existing and non-existing Formatters for each Protocol | |
| **Contracts:**  20. Scan PDML  *Identify Protocols*  identifyProtocols(PDML)  Description: Receives a PDML file and returns a list of all identified Protocols from it.  Pre:  Post: The PDML file is not modified.  *Identify Formatters*  identifyFormatters(PDML)  Description: Receives a PDML file and returns a list of all applied Formatters from it.  Pre: The File has been opened by PFS.  Post: The PDML file is not modified.  *Create Formatter*  Create Formatter(Protocol)  Description: Receives a Protocol for which to build a new Formatter.  Pre:  Post: An empty set of Rules is created for this Formatter.  **Private Responsibilities:** | **Collaborations:**  Formatter (10)  Formatter (10)  Formatter (10) |
| **Comments:**  Reads through Protocols in a modifiable PDML | |

|  |  |
| --- | --- |
| **Class Name:** Rule | |
| **Description:** Set of Filters and Actions which to build the Formatter to modify the Protocol of a PDML | |
| **Contracts:**  22. Manage Rule  *Create Rule*  createRule()  Description: Creates the structure by which to contain Actions for Filters  Pre:  Post:  *Add Filter*  addFilter(Filter)  Description: Adds a Filter for a set of Actions  Pre:  Post: Appends to the list of Filters in the Rule structure  *Delete the Last Filter from Rule*  deleteFilter()  Descritpion: Deletes the last Filter added to this Rule  Pre:  Post:  23. Manage Rule’s Actions  *Add a New Action to Rule*  addAction(Action)  Description: Receives an Action and adds it to the Rule.  Pre:  Post: Appends to the list of Actions in the Rule structure  *Delete the Last Action from Rule*  deleteAction()  Descritpion: Deletes the last action added to this set of Actions for the Rule  Pre: The Action is the last added Action to the Rule.  Post:  **Private Responsibilities:** | **Collaborations:**  Filter (8, 9)  Filter (8)  Action (1), Hook (13)  Action (1), Hook (13) |
| **Comments:**  Contains Filters and Actions | |

|  |  |
| --- | --- |
| **Class Name:** Script | |
| **Description:** Subtype of File Class which contains the Python algorithm by which to update Field(s) | |
| **Contracts:**  24. Manage Script  *Create a New Script*  createScript(Hook)  Description: Creates a Script for this Hook  Pre:  Post: The Script Python File will be created if it does not already exist.  *Delete Script*  deleteScript(Hook)  Description: Deletes the Script from this Hook  Pre: The Script is associated with this Hook.  Post: The Script File is not deleted.  25. Modify Script  *Update Algorithm*  updateAlgorithm(String<>)  Description: Overwrites the existing algorithm to contain this String<>  Pre: The Script Python File exists.  Post:  *Update File Path*  updatePath(String)  Description: Sets the path to this Script to be this String  Pre: This String is a valid file system path to this Algorithm  Post: The Script is not deleted.  26. Run Script  *Get Return Value*  getReturnValue()  Description: Gets the return value from the Script’s execution  Pre: The Script has completed execution and has a return value  Post:  *Update Field*  updateField(Field)  Description: Updates the Field associated with this Hook  Pre: The Script’s return value type matches the Field’s value type  Post:  **Private Responsibilities:** | **Collaborations:**  Hook (13, 14)  Hook (14)  Hook (14)  Hook (14)  Hook (15)  Hook (15) |
| **Comments:**  Contains editable algorithm and file path; type of File | |

|  |  |
| --- | --- |
| **Class Name:** tShark | |
| **Description:** Handles operations for communications between tShark and Formatter Protocol System (FPS) | |
| **Contracts:**  27. Convert PCAP to PDML  *Send PCAP File*  sendPCAP(PDML)  Description: Receives a PCAP file to be parsed by tShark tool  Pre: The file has not been stored before on the current local filesystem of the user.  Post: tShark returns no error on accepting the PDML file to parse.  *Return Parsed PDML File*  storePDML()  Description: Receives a parsed PDML file and stores it on the local filesystem for PFS to access.  Pre: tShark parsed the PDML with no errors.  Post: Any file with the same name will be overwritten.  **Private Responsibilities:** | **Collaborations:**  File (7)  File (7) |
| **Comments:**  External tool to Formatter Protocol System (FPS) | |