УДК 683.01

АВТОМАТИЗАЦИЯ КОНТРОЛЯ ПРИМЕНЕНИЯ СТИЛЯ ЯЗЫКА ПРОГРАММИРОВАНИЯ

Ю.М. Крамар

Национальный авиационный университет,
03058, Киев, 58, проспект Космонавта Комарова, 1, т. 484-96-41
e_mail: samix@nbi.com.ua

Приводится описание процесса контроля применения стиля языка программирования и устройства, автоматизирующего данный процесс, описывается математическая модель устройства и алгоритм его функционирования. Представляются архитектура средств, реализующих контроль применения стиля, и результаты экспериментальных исследований программных средств, автоматизирующих контроль применения стиля языка программирования.

The description of the programming language style using control and device, that automates control, of the device mathematical model and its functioning algorithm. The style using automates control means architecture and software experimental researches results are represented.

Введение

В связи с использованием инженерных методов разработки программного обеспечения (ПО), повторного использования и экстремального программирования при корпоративной разработке ПО особое значение приобретает использование стилей программирования при написании текстов программ [1, 2, 3]. Под стилем программирования будем понимать стиль, используемый в деятельности человека (домене), сущность которой состоит в создании программ для компьютеров [4]. Стиль программирования, представленный в определенном языке программирования, называется стилем языка программирования. Его использует программист в качестве инструмента для написания программ [4].

Для того чтобы применение стиля программирования при разработке ПО было обязательным, необходим контроль его применения, поддерживаемый соответствующими средствами. Средства поддержки применения стиля программирования могут быть трех типов: редакторы исходных текстов (например, редактор текстов в среде программирования Turbo Pascal, редактор утилиты QuickDesk); среды программирования (например, C Builder, Delphi, Visual Studio .NET); специальное ПО (Lint [4], PVE Лаборатории Касперского).

Редакторы исходных текстов обеспечивают представление стиля, поддерживая некоторые правила. Например, редакторы, встроенные в среды программирования Turbo Pascal, C Builder, Delphi, автоматически расставляют отступы горизонтального типа при переходе на новую строку. Шаг отступа устанавливается редактором или пользователем. Редактор утилиты QuickDesk, предназначенной для создания текстов хранимых процедур и триггеров, поддерживает определенную форму комментариев и некоторых операторов. Так при наборе первого ключевого слова (if, declare, while) конструкции оператора редактор автоматически дописывает всю конструкцию, оставляя пользователю возможность вставить обозначение, тип переменной, выражение, тело оператора или комментарий.

В средах программирования при автоматической генерации кода соблюдаются правила составления идентификаторов, рекомендуемые Microsoft, правила расстановки отступов, принятые в структурном программировании [1], некоторые правила стиля для объектно-ориентированного программирования. Например, при автоматической генерации описания интерфейса класса каждый спецификатор доступа используется один раз, а элемент класса описывается отдельной строкой. При генерации метода класса описание метода выделяется пустыми строками (C Builder, Delphi) и специальными комментариями (C Builder), обозначения методов составляются в смешанном регистре (например, ImageMouseMove, ImageDblClick, FormShow), длина строки не превышает восьмидесяти символов.

К специальному ПО относится PragueVisualEditor (PVE), разработанное «Лабораторией Касперского» и использующееся им для поддержки применения корпоративного стиля.

Программисты коллектива разработчиков пишут программы с помощью данного редактора, чем обеспечивается полное соблюдение выработанного корпоративного стиля программирования. Например, для описания новой константы программист указывает в соответствующих полях диалоговых окон редактора ее обозначение, тип, значение, краткое и полное описание назначения константы, а также описание поведения константы в различных условиях. Система автоматически генерирует соответствующий исходный код с комментариями. Аналогичные действия программист выполняет при описании пользовательских типов, операторов, списков включаемых заголовочных файлов и списков кодов ошибок, возвращаемых методами классов интерфейса. Недостатком такого средства является невозможность изменения или корректировки «зашитого» в них стиля программирования. Кроме того, средство нельзя использовать при обучении

программированию, так как программист-новичок, не создавая сам исходный текст, не способен разобраться в тексте программы и освоить изучаемый материал.

Специальное ПО Lint отслеживает применение некоторых стилевых рекомендаций в текстах программ, а также указывает на участки кода, где в процессе эксплуатации программного продукта могут возникнуть сбои. В средстве Lint, например, также входит небольшое описание рекомендаций, соблюдение которых он контролирует. Недостатком этой программы является отсутствие возможности изменить состав контролируемых рекомендаций. Фактически Lint, и подобные ей программы «навязывают» пользователю стиль программирования. Имеется еще несколько коммерческих программных средств проверки программ, написанных на C++ [5], но их функционирование, в основном, направлено на оценивание некоторых показателей качества (эффективность, надежность, устойчивость к сбоям).

Существуют также утилиты «красивая печать» [6] (например, indent), которые форматируют текст программы. Однако они являются средствами, скорее преобразующими стиль программы, чем контролирующими применение стиля языка программирования.

Постановка проблемы

Анализ средств, поддерживающих применение стиля путем автоматизации процесса контроля, показывает, что средства обладают существенными недостатками в аспектах поддержки стиля (слабый контроль в редакторах и средах программирования), возможности модифицировать стиль («навязывание» правил Lint) и области применения (PVE невозможно использовать при обучении). Кроме того средства поддержки применения стиля автоматизируют отдельные процессы применения стиля, и совместное их использование не возможно. Например, применение стиля, «зашитого» в PVE, не удается контролировать с помощью Lint, и не возможно преобразовать, используя indent.

Поэтому существует необходимость в создании средств, комплексно автоматизирующих создание стиля языка программирования и контроль его применения.

Статья состоит из трех частей. В первой части приводится описание процесса контроля применения стиля языка программирования и математической модели устройства, автоматизирующего данный процесс, а также алгоритм функционирования устройства. Во второй части рассматривается архитектура средств, реализующих контроль применения стиля, и в третьей части представляются результаты экспериментальных исследований программных средств, автоматизирующих контроль применения стиля языка программирования.

Решение проблемы

Контроль применения стиля языка программирования. Применение стиля языка программирования включает в себя процессы создания стиля и контроля его применения. Процесс создания стиля языка программирования, а также устройство, автоматизирующее данный процесс, описаны в работе [7].

Контроль стиля языка программирования – это процессы проверки применения стиля языка программирования при написании текстов программ. Необходимость выполнения процессов возникает, когда требуется определить соответствие стиля программы заданному стилю языка программирования разрабатываемого ПО, а также при обучении написанию программ.

Для автоматизации процесса контроля сформулируем соответствующую задачу, разработаем математическую модель для ее решения и определим устройство, решающее поставленную задачу и автоматизирующее данный процесс. Кроме того, для реализации процесса контроля необходимы формальные описания стиля программирования и стиля языка программирования, а также процессов их создания [7, 8, 9].

Сущность решения задачи контроля состоит в том, чтобы определить, соответствует ли текст программы T, написанный на языке программирования L, определенному стилю S_{Lx} языка программирования. Так как стиль языка программирования представляется множеством правил r_{Li}, то при решении задачи контроля стиля необходимо определять, соблюдались ли правила, входящие в стиль языка программирования.

Математическую модель объекта моделирования можно представить в виде множества величин, описывающих процесс функционирования реального объекта и образующих следующие подмножества: совокупность входных воздействий $x_i \in X, i = \overline{1, n_X}$; совокупность воздействий внешней среды $v_l \in V, l = \overline{1, n_V}$; совокупность внутренних (собственных) параметров $h_k \in H, k = \overline{1, n_H}$; совокупность выходных характеристик $y_j \in Y, j = \overline{1, n_H}$. В общем случае процесс функционирования моделируемого объекта описывается во времени законом функционирования моделируемого объекта, представленным оператором f, который преобразует экзогенные переменные (x_i v_l, h_k) в эндогенные (y_j) в соответствии с соотношением вида $Y(t) = f(X, V, H, t)$. Метод получения выходных характеристик Y с учетом входных воздействий X, воздействий внешней среды V и собственных параметров H называется алгоритмом функционирования модели.

Разрабатываемая математическая модель является детерминированной и статической, так как стохастические переменные V и H и переменная t отсутствуют, тогда оператор f имеет следующий вид: $Y = f(X)$, где X – входные характеристики, Y - выходная. Таким образом, в данном случае, чтобы

определить модель объекта, необходимо указать конечное множество X вместе с математическими связями между ним и характеристикой Y [10].

В зависимости от целей, для достижения которых предполагается использовать результаты решения задачи контроля, задачу следует решать двумя способами. Первый способ позволяет установить факт соблюдения или несоблюдения всех правил множества R_{Lx} стиля S_{Lx} при написании текста программы T. Второй способ позволяет определять величины, указывающие степень соблюдения правил множества R_{Lx} стиля S_{Lx} при написании текста программы T.

Решение задачи контроля стиля можно осуществить следующим образом. Существует некоторое устройство, реализующее процессы контроля стиля. На его вход поступает текст программы T, информация о стиле S_{Lx} и о способе контроля D_c. На выходе устройства получается значение функции F (рис. 1).

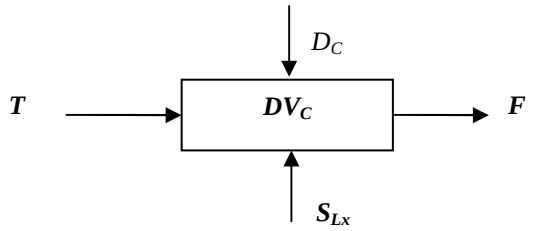

Рисунок 1. Схема устройства контроля стиля

Для решения задачи контроля первым способом (D_c = hard) опишем логическую функцию $f_1(r_{Li}, T)$, где r_{Li} – правило, T – текст программы. При этом, если правило r_{Li} соблюдается при написании текста программы T (количество случаев применения правила равняется количеству случаев, в которых возможно применение правила), то $f_1(r_{Li}, T)=1$, иначе $f_1(r_{Li}, T)=0$.

Тогда будем считать, что текст программы T соответствует стилю S_{Lx}, если выполняется следующее условие: $\forall r_{Li}(f_1(r_{Li},T)=1)$, $r_{Li} \in R_{Lx}$. Сущность решения задачи контроля стиля состоит в получении значения функции $F=F_1$:

$$F_1 = f_1(r_{L1},T) \wedge f_1(r_{L2},T) \wedge ... \wedge f_1(r_{LI},T), \qquad I = |R_{Lx}|.$$

При этом, функция F_1 принимает значение единица, если для каждого правила r_{Li} функция f_1 возвращает единицу, и получает значение нуль, если существует хотя бы одно правило r_{Li}, для которого f_1 возвращает нуль.

Для решения задачи контроля вторым способом (D_c = soft) будем использовать функцию f_2 (r_{Li}, T) и удельный вес q_i правила r_{Li}, характеризующий степень важности соблюдения правила при использовании стиля S_{Lx} в написании текста программы T. Функция f_2 (r_{Li}, T) принимает значения [0; 1] и является отношением количества случаев применения правила r_{Li} в тексте T к количеству случаев, в которых возможно применение правила. Тогда решением задачи контроля является получение значения функции $F=F_2$:

$$F_2 = f_2(r_{L1},T)*q_1 + f_2(r_{L2},T)*q_2 + ... + f_2(r_{LI},T)*q_I, \qquad I = |R_{Lx}|.$$

Функция F_2 принимает значения от нуля до единицы, в зависимости от количества соблюдаемых правил множества R_{Lx} стиля S_{Lx} в тексте программы T, а также от их весов Q_x.

Контролируемые тексты T, информацию о стиле S_{Lx} и о способе контроля D_C определим как совокупность входных воздействий модели. Функцию F определим как выходную характеристику модели. Тогда закон функционирования системы представляется следующими математическими отношениями:

$$X = (T, S_x, D_C); \quad Y = F = \begin{cases} F_1, & если \quad Dc = hard \\ F_2, & если \quad Dc = soft \end{cases}.$$

Функционирование модели первым способом описывается следующим алгоритмом:

1. устанавливается значение функции F_1 в единицу;
2. отбирается следующее по порядку, начиная с первого, правило r_{Li}, входящее в стиль S_{Lx}; если отбор правил закончен, выполняется переход к п. 5;
3. вычисляется промежуточное значение функции F_1: $F_1 = F_1 \wedge f_1(r_{Li},T)$;
4. если $F_1 = 0$, делается вывод о несоответствии текста программы T стилю программирования S_{Lx} и решение задачи контроля стиля прекращается; если $F_1 = 1$, выполняется переход к п. 2;
5. если конечное значение функции равно единице, делается вывод о соответствии текста программы T стилю программирования S_{Lx}.

Функционирование модели вторым способом описывается следующим алгоритмом:

1. устанавливается значение функции F_2 в нуль;

2. отбирается следующее по порядку, начиная с первого, правило r_{Li}, входящее в стиль S_{Lx}; если отбор правил закончен, выполняется переход к п. 5;

3. определяется значение функции f_2 (r_{Li}, T);

4. если f_2 (r_{Li}, T)>0, то значение f_2 (r_{Li}, T)*q_i добавляется к текущему значению функции F_2; выполняется переход к п. 2;

на основании конечного значения функции F_2 делается вывод о степени соблюдения правил множества R_{Lx} стиля S_{Lx} при написании текста программы T.

Архитектура программных средств, автоматизирующих контроль применения стиля языка программирования. Программные средства состоят из следующих подсистем (рис. 2): контроля соблюдения правила; вычисления функций $f_1(r, T)$ и $f_2(r, T)$; формирования результатов "жесткого" и "мягкого" контроля.

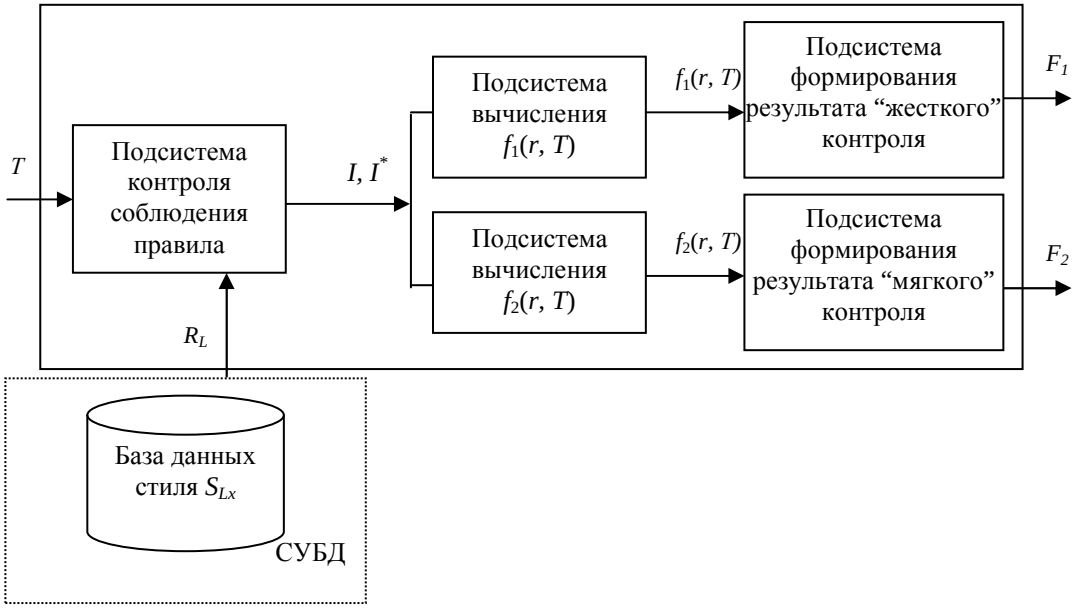

Рисунок 2. Архитектура средств контроля применения стиля

База данных стиля языка программирования содержит описание правил стиля (пассивную часть) и средства определения условий соблюдения правил в текстах программ (активную часть) [7, 8].

Подсистема контроля соблюдения правила служит для определения значения I, указывающего, сколько раз было применено правило стиля r_L, и I^*, указывающего возможность применения правила. Сущность работы подсистемы состоит в следующем:

- в главной форме программного средства указывается название контролируемого стиля, тип контроля, конфигурация программы (путь к пассивной части БД стиля, директория с активной частью и директория для создания временных рабочих файлов), выбирается список файлов, содержащих контролируемые исходные тексты программ, и дается команда начала контроля;
- при отсутствии ошибок ввода последовательно считываются из БД стиля ссылки пассивной части на активную, находятся программы-измерители и запускаются на выполнение для каждого контролируемого исходного текста;
- после завершения работы измерителя, результаты его выполнения считываются из рабочего файла result.tmp и передаются подсистеме вычисления $f_1(r, T)$ или подсистеме вычисления $f_2(r, T)$ в зависимости от выбранного типа контроля.

Подсистема вычисления $f_1(r_L, T)$ определяет значение функции f_1:

$$f_1(r_L, T) = \begin{cases} 0, & если \quad I < I* \\ 1, & если \quad I = I* \end{cases}.$$

Сущность работы подсистемы состоит в следующем:

- определяется значение функции f_1;
- результат записывается в файл статистики для отображения в форме просмотра статистики контроля стиля;
- значение функции f_1 передается подсистеме формирования результата "жесткого" контроля.

Подсистема формирования результата "жесткого" контроля предназначена для вычисления функции F_1 и представления результата контроля пользователю. Сущность работы подсистемы состоит в следующем:

- определяется значение функции F_1: $F_1 = F_1 \wedge f_1$;

- если $F_1 = 0$, формируется отрицательный результат “жесткого” контроля, и выдается сообщение о несоответствии текстов программ T заданному стилю S_{Lx}, иначе процесс контроля продолжается;
- если после проверки всех исходных текстов программ на соблюдение правил стиля $F_1 = 1$, формируется положительный результат “жесткого” контроля, и выдается сообщение о соответствии текстов программ T заданному стилю S_{Lx}.

Подсистема вычисления $f_2(r, T)$ предназначена для определения значения функции f_2:

$$f_2(r_L, T) = \begin{cases} \dfrac{I}{I^*}, & \textit{если} \quad I^* > 0 \\ 1, & \textit{если} \quad I^* = 0 \end{cases}.$$

Сущность работы подсистемы аналогична работе подсистемы вычисления $f_1(r, T)$, результаты вычисления передаются подсистеме формирования результата “мягкого” контроля.

Подсистема формирования результата “мягкого” контроля предназначена для вычисления функции F_2, представления результата контроля пользователю и формирования файла статистики. Не зависимо от того, какое значение принимает функция $f_2(r, T)$, средства контроля стиля в режиме “мягкого” контроля осуществляют полную проверку всех исходных текстов программ на соблюдение правил стиля. В главной форме отображается результат “мягкого” контроля, и выдается сообщение о степени соответствия текстов программ T заданному стилю S_x. Подробная информация о результатах “мягкого” контроля представляется в форме просмотра статистики контроля стиля.

Результатом работы средств является оценка применения стиля программирования S_{Lx} при написании текстов программ T, представленная значением функции F_1 или F_2 в зависимости от выбранного вида контроля.

База данных стиля языка программирования получена в процессе создания стиля языка программирования с помощью программных средств создания стилей программирования и стилей языка программирования [7, 8]. Для ее реализации в качестве СУБД использовалась система INTERBASE 6.0. Программные средства, автоматизирующие процессы создания стилей и контроль применения стиля языка программирования, реализованы на языках Visual C, C++ и Delphi и представляются клиентскими приложениями Style.exe и Control.exe.

Исследования программных средств, автоматизирующих процесс контроля применения стиля языка программирования. Описанные в работе средства контроля применения стиля языка программирования, применяются в комплексе со средствами, автоматизирующими другие процессы применения стиля программирования (создание стиля программирования и стиля языка программирования, анализ и преобразование стиля программы).

Комплекс можно использовать в разработке ПО и обучении программированию. Для каждой области необходимо разработать информационную технологию, в основе которой будут программные средства, автоматизирующие процессы применения стиля программирования. Эффективность информационной технологии определяется повышением эффективности процессов, которые она автоматизирует. Информационная технология должна отвечать предъявляемым к ней требованиям, которые специфичны для каждой области. В качестве области, в которой исследовалась работоспособность программных средств, выбрано обучение программированию.

В результате анализа предметной области рассмотрены особенности обучения программированию, которые позволяют выявить проблемы контроля неавтоматизированным способом (оперативность и объективность контроля), а также автоматизированным способом (достоверность и простота взаимодействия) и выработать соответствующие требования, предъявляемые к программным средствам.

При этом требования достоверности и объективности являются главными, которым должны отвечать любые средства, автоматизирующие процессы человеческой деятельности. Поэтому, процесс контроля отвечает данным критериям эффективности, если выполняется автоматизировано, с применением разработанных средств, дающих достоверные и объективные результаты. Оперативность и простота взаимодействия являются требованиями, специфичными для исследуемой области. Эти критерии процесса можно определить, используя соответствующие критерии для средств, автоматизирующих процесс. Затем для автоматизации процессов применения стилей программирования в обучении, предложена информационная технология, в которую входят разработанные средства контроля

Для проверки решений, принятых при разработке средств, был запланирован и проведен ряд экспериментов. Целью экспериментов являлось выполнение процессов применения стиля с помощью разработанных средств, изучение результатов выполнения для исследования свойств средств, используемых в обучении, и оценки выполнения предъявленных требований. Для этого в результате выполнения процессов создания были получены БД стилей языка программирования STYLE_1 и STYLE_2. Участникам эксперимента были представлены описания двух созданных стилей, и предлагалось выбрать один из них для применения в программировании. Тексты программ студентов, выбравших стиль STYLE_1, составили первую группу, а тексты студентов, выбравших стиль STYLE_2, - вторую группу. Совместно со стилем студентами для изучения использовалась система представления стиля пользователю.

Программы первой и второй группы были подвергнуты контролю текстов на соответствие заданному стилю. Контроль текстов студенческих программ выполнялся в «мягком» и «жестком” режимах контроля. Кроме студенческих программ в каждую из групп была включены контрольные программы, полностью отвечающие соответствующему стилю, и программы, наоборот, не отвечающие стилю. После выполнения процесса контроля стиля, все тексты были объединены в одну группу, и был проведен процесс анализа текстов на соответствие определенному стилю.

Для определения качественной оценки критерия достоверности были получены результаты работы средств, автоматизирующих контроль применения стиля языка программирования и анализ стиля программы, и проведен сравнительный их анализ.

В процессе сравнительного анализа результатов работы систем контроля и анализа стиля программирования выявлены следующие особенности:

1. контрольные программы, полностью отвечающие стилю, по результатам «жесткого» контроля попали в группу, где стиль соблюден, составляющую 1%.
2. контрольные программы, не отвечающие стилю, по результатам «жесткого» контроля попали в группу, где стиль не соблюден, составляющую 99%.
3. Программы, в которых стиль соблюден не полностью, по результатам «жесткого» контроля также попали в группу, составляющую 99%.
4. Набор программ, в которых по результатам «мягкого» контроля стиль соблюдается более чем на половину, шире набора программ, отвечающих стилю при «жестком» контроле. При этом, программы второй группы составляют подмножество программ первой группы.
5. Программы, в которых по результатам «мягкого» контроля стиль соблюдается более чем на половину, при анализе их текстов на соответствие стилю вошли в группу, составляющую 65%, где стиль определен однозначно.

Программы, в которых по результатам «мягкого» контроля стиль соблюдается менее чем на половину, вошли в группы, где стиль определен неоднозначно (15%) или не определен (20%).

Полученные данные доказывают, что результаты работы системы анализа стиля не противоречат результатам работы системы контроля стиля и результатам, полученным при выполнении процессов вручную. Поэтому был сделан положительный вывод о достоверности результатов, полученных средствами, автоматизирующими процессы применения стиля программирования. Зависимость достоверности оценки от количества правил, составляющих стиль, представлена на графике (рис. 3).

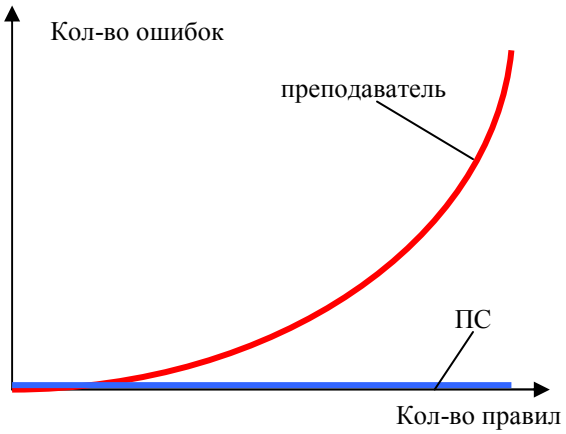

Рисунок 3

Объективность результатов автоматизированного контроля обеспечивается отсутствием человеческого фактора и использованием точного математического аппарата для формирования оценки знаний.

Для оценки оперативности средств, автоматизирующих процессы применения стиля программирования, в эксперименте определялось время работы средств в процессе контроля программ на соответствие стилю. Эффективность применения средств по критерию оперативности состоит в том, что время контроля стиля, производимого автоматизированным способом с помощью предложенных средств, не выходит за установленный предел и позволяет увеличить время контроля материала дисциплины. Зависимость времени контроля от количества правил стиля и достоверности оценки от времени контроля представлены на графиках (рис. 4, 5).

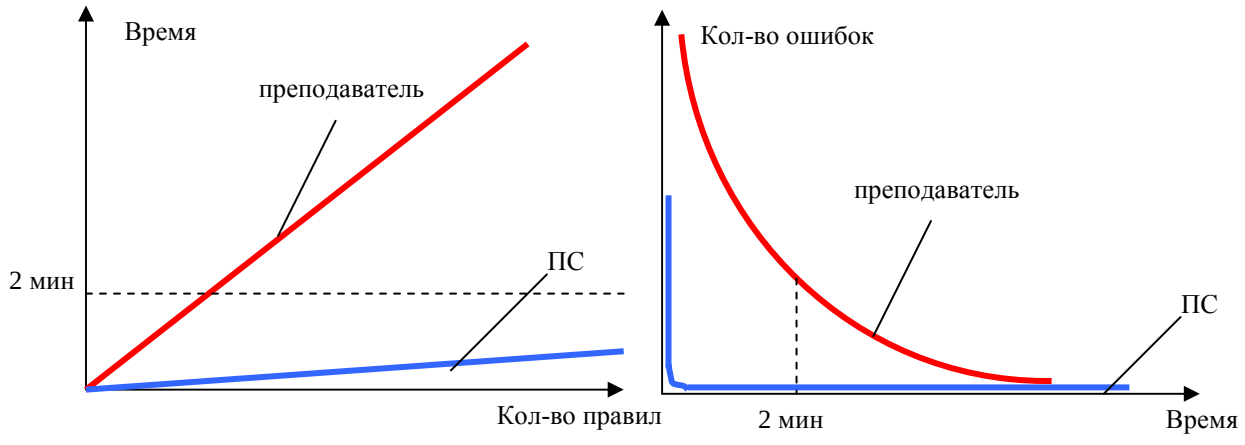

Рисунок 4 Рисунок 5

Оценка простоты взаимодействия со средствами, автоматизирующими процессы применения стиля программирования, - это «процесс, в котором оценивается удобство использования интерфейса и степень его соответствия требованиям пользователя» [11]. Оценивание производилось, по таким показателям удобства использования интерфейса как изучаемость, скорость работы, устойчивость, восстанавливаемость и адаптируемость [11]. При этом, для каждого из показателей была получена качественная оценка.

Заключение

Описанный в статье процесс контроля применения стиля языка программирования является одним из процессов применения стиля в программировании, а предложенные программные средства составляют комплекс программных средств, автоматизирующих процессы применения стиля программирования при разработке ПО и обучении программированию. Анализ результатов оценки критериев процесса контроля обучения и средств, автоматизирующих процессы применения стилей программирования, показал, что применение этих средств позволяет сделать процесс контроля обучения более эффективным, повысив качество обучения программированию и связанным с ним дисциплинам.

Литература

1. *Керниган Б., Плоджер Ф.* Элементы стиля программирования. - М.: Радио и связь, 1984. – 160 с.
2. *Ален И., Голуб А.* С и С++. Правила программирования. – М., БИНОМ. – 272 с.
3. *Нуквист Е.* Правила хорошего тона для программирования на С++. - Киев: Наук. думка, 1994. – 85 с.
4. *Сидоров М.О.* Основні поняття стилістики програмування. – Проблеми інформатизації та управління: збірник наукових праць: випуск 9. – К.: НАУ, 2003. – с. 318-320.
5. *Мейерс С.* Эффективное использование С++. 50 рекомендаций по улучшению программ и проектов. – М.: ДМК, 2000. – 240 с.
6. *Боровин Г. К.* Ошибки-ловушки при программировании на Фортране. – М.: Наука, 1987. – 144 с.
7. *Крамар Ю. М.* Автоматизация решения задач стилистики программирования// Проблеми інформатизації та управління: збірник наукових праць: випуск 5. – К.: НАУ, 2002. – с. 211-215.
8. *Крамар Ю. М.* Средства для автоматизированного синтеза стилей программирования// Вестник НАУ №2 (13) 2002. – К.: НАУ, 2002. – с. 52 - 60.
9. *Крамар Ю. М.* Схема систематизации и представление правил стилей программирования// Проблемы программирования. № 1-2. - К.: - 2002 р., с.
10. *Советов Б. Я., Яковлев С. А.* Моделирование систем. – М.: Высш. шк., 2001. – 343 с.
11. *Соммервилл И.* Инженерия программного обеспечения. – М.: Издательский дом “Вильямс”, 2002. – 624с.