Перчёнок Олег Владимирович,
Клионский Дмитрий Михайлович

УДК 004.434

АВТОМАТИЗАЦИЯ ВЕРИФИКАЦИИ РЕШЕНИЯ ЗАДАЧ В ОБЛАСТИ ЦИФРОВОЙ ОБРАБОТКИ СИГНАЛОВ

Аннотация

В статье рассматривается подход к автоматизации верификации задач в области цифровой обработки сигналов с использованием предметно-ориентированного языка системы WiseTasksDSP, которая разрабатывается авторами статьи. Демонстрируется пример, связанный с нахождением верхней граничной частоты спектра дискретного сигнала и проверки частоты дискретизации на ее допустимость для точного (без потерь информации) восстановления непрерывного сигнала из дискретного. Кроме того, упоминаются другие задачи, в которых применение системы WiseTasksDSP может оказаться эффективным.

Ключевые слова: предметно-ориентированный язык, цифровая обработка сигналов, Java, MATLAB.

ВВЕДЕНИЕ

В статье рассматривается подход к автоматизации верификации задач в области цифровой обработки сигналов с использованием предметно-ориентированного языка системы **WiseTasksDSP,** которая разрабатывается авторами статьи. Демонстрируется пример, связанный с нахождением верхней граничной частоты спектра аналогового сигнала и проверки частоты дискретизации дискретного сигнала на ее допустимость для точного (без потерь информации) восстановления непрерывного сигнала из дискретного. Кроме того, упоминаются другие задачи, в которых применение системы WiseTasksDSP может оказаться эффективным.

Предметно-ориентированный язык – это язык программирования, предназначенный для решения задач в определенной предметной области. Языки данного типа позволяют сделать некоторые частные задачи программирования более легкими для понимания программистов, а также для специалистов предметной области. Благодаря этому программы на предметно-ориентированных языках можно быстрее писать, легче изменять, и они менее подвержены ошибкам, чем программы на языках общего назначения (таких, как C++, Pascal, Java). Кроме этого, упрощается взаимодействие между специалистами предметной области и программистами. Написание программ на некоторых предметно-ориентированных языках вообще не требует вмешательства программистов (например, языки файлов конфигурации пакетов программ в формате XML или INI) [1].

В системах автоматизированного контроля знаний, применяемых в российских учреждениях высшего профессионального образования, обычно используется принцип тестирования: студенту предлагается несколько вариантов ответа, из которых он выбирает один или несколько. При этом тестирующую систему не интересует алгоритм решения, которым пользовался студент. Он может наугад выбрать вариант ответа, который, если «повезет», окажется правильным. В описываемой в статье системе WiseTasksDSP невозможно подобрать правильный ответ, так как в ней верифицируется алгоритм решения.

В работе [2] описана система PCMS, применяемая для проверки решений задач на олимпиадах по программированию. В системе PCMS верификация программы-решения заключается в прохождении ряда тестов, заключающихся в подаче на вход программы заранее заготовленного эталонного файла исходных данных и сравнение получившегося файла выходных данных с соответствующим эталонным выходным файлом. Недостаток данного метода заключается в том, что необходимо хранить эталонные файлы. В системе WiseTasksDSP в этом нет необходимости, так как проверка осуществляется при помощи специальной функции-верификатора, в которой в цикле меняются исходные данные к задаче, запускается решение студента и осуществляется проверка полученных выходных данных.

Ранее проводилась разработка систем электронного обучения в соответствии с обозначенным подходом. В качестве примера можно привести систему «Конструктор комбинаторных коллекций», обеспечивающую автоматическую верификацию решений задач по комбинаторике на основе формального описания условия [3], а также систему WiseTasksGeometry[1], осуществляющую аналогичную поддержку задач на построение по геометрии [4, 5]. В разработке последней системы принимал участие один из авторов статьи. Система включает модули учителя и ученика, реализованные на принципиально разных концепциях. В модуле учителя автор задачи описывает условие в виде текста и в виде предикатов и логических связок между ними в привычной для себя форме, строит исходный чертеж и задает набор инструментов, которым сможет пользоваться ученик при решении задачи. В модуле ученика школьник может прочитать текстовое условие, выполнить необходимые построения и отослать решение на контроль, который состоит в проверке заданных учителем предикатов. Построение чертежей осуществляется с помощью свободно распространяемой системы динамической геометрии GeoGebra [6].

[1] Wise tasks – умные задачи (англ.).

1. МОДЕЛЬ САМОПРОВЕРЯЕМОЙ ЗАДАЧИ

Авторы статьи участвуют в разработке систем электронного обучения, в которых описание задач строится на формальной модели предмета и порождает адекватные формальные описания задач, поддающиеся автоматической обработке. Название таких систем начинается с «WiseTasks…» (англ. «умные задачи»). Решение задачи осуществляется студентом (или учеником) в предметной среде на основе инструментов, являющихся существенными для данного класса задач. Оно верифицируется системой на формальном описании задачи и не требует предварительного решения задачи автором или проверяющим.

Определение 1. Описанием условия самопроверяемой задачи называется пара

$$P = \langle T, F \rangle,$$

где T – неформальное условие задачи (например: текст, изображения); F – формальное условие задачи, $F \in \Psi$, где Ψ – множество всех возможных формальных условий задачи данного класса.

Определение 2. Верификатором задачи называется предикат

$$v\colon \Psi \times D \to \{false, true\},$$

где D – множество всех возможных решений задачи данного класса; Ψ – множество всех возможных формальных условий задачи данного класса.

Верификатор принимает значение *true*, если задача решена верно.

Определение 3. Описанием класса самопроверяемых задач называется кортеж

$$C = \langle L_P, L_S, \Omega, \Omega', v \rangle,$$

где L_P – предметно-ориентированный язык описания условия задач данного класса; L_S – предметно-ориентированный язык описания решения задач данного класса; Ω – множество всех возможных инструментов решения задач данного класса; Ω' – множество инструментов, по умолчанию доступных для решения задач данного класса, $\Omega' \subset \Omega$; $v\colon \Psi \times D \to \{false, true\}$ – верификатор задачи.

Абстрактная модель, на которой основан предметно-ориентированный язык описания условия задачи L_P, изображена на рис. 1 в виде UML-диаграммы классов [7]. Описание задачи состоит из заголовка и трех списков: неформальных условий, ограничений на множество инструментов, доступных при ее решении, и верификационных условий.

Неформальные условия – это информация, которую будет видеть ученик (студент), решающий задачу. В зависимости от предметной области такие условия могут быть объектами разных типов. Например, текстового (текстовое условие задачи), графического (поясняющая картинка или схема). Кроме этого, такое условие может быть внедренным объектом из других приложений (например, формула MathType).

Список ограничений на множество инструментов может быть пустым. Если ограничений на множество инструментов нет, то для решения данной задачи предоставляются инструменты из множества Ω' данного класса задач (см. определение 3). Ограничения могут не только уменьшать множество инструментов за счет запрещения использования некоторого подмножества Ω', но и разрешать использовать дополнительные инструменты из множества Ω / Ω'.

В описании задачи должно быть хотя бы одно верификационное условие. Такие условия позволяют вычислить значение верификатора для конкретной задачи. Верификационное условие может быть представлено в виде предиката или может задавать какое-то состояние среды для проверки условий (например, исходное множество для задач по комбинаторике или исходный чертеж для задач по геометрии).

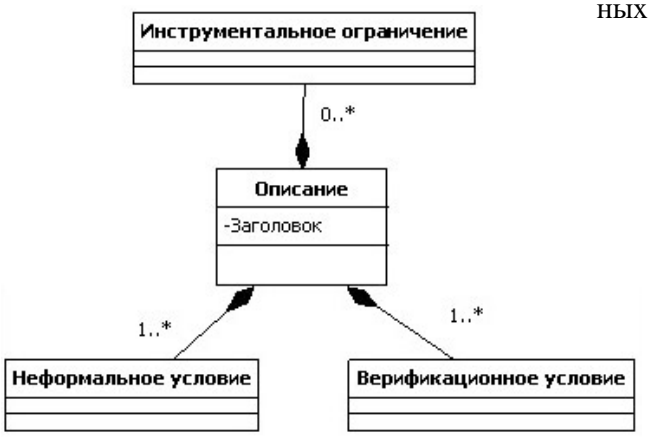

Рис. 1. Абстрактная модель, на которой основан предметно-ориентированный язык описания условия задачи L_P

В рамках каждой предметной области на основе абстрактной модели строится конкретная модель для предметно-ориентированного языка.

На основе абстрактной модели можно построить абстрактную грамматику языка, которая, будучи записанной в форме Бэкуса-Наура [1], имеет следующий вид:

```
Описание ::= Заголовок {НеформальноеУсловие}
        {ИнструментальноеОграничение}
        {ВерификационноеУсловие}
```

В данной грамматике все символы являются нетерминальными. Конкретизация абстрактной грамматики выполняется для определенной предметной области.

Структура предметно-ориентированного языка описания решения задач L_S зависит от предметной области. Единственное условие – обязательное отражение в языке инструментов из множества Ω, что необходимо для проверки соблюдения инструментальных ограничений при верификации задачи.

Процедура верификации задачи заключается в следующем:

1. Проверка выполнения инструментальных ограничений;
2. Проверка выполнения верификационных условий.

2. ВЗАИМОДЕЙСТВИЕ СИСТЕМЫ WISETASKSDSP И MATLAB

Одно из приложений модели самопроверяемых задач – система WiseTasksDSP (DSP – Digital Signal Processing, ЦОС). Разрабатываемая система основана на взаимодействии с инженерно-математическим пакетом MATLAB [8].

MATLAB (сокращение от англ. *«Matrix Laboratory»*) — термин, относящийся к пакету прикладных программ фирмы Mathworks Inc. для ре-

шения задач при помощи технических вычислений и имитационного моделирования, а также к используемому в этом пакете языку программирования.

Язык MATLAB является высокоуровневым интерпретируемым языком программирования, включающим основанные на матрицах структуры данных, широкий спектр функций (как стандартных, так и внешних, хранящихся в библиотеках), интегрированную среду разработки, объектно-ориентированные возможности и интерфейсы к программам, написанным на других языках программирования.

Программы, написанные на MATLAB, бывают двух типов – функции (файл-функции) и сценарии (скрипты, англ. scripts). Функции имеют входные и выходные аргументы (параметры), а также собственное рабочее пространство для хранения промежуточных результатов вычислений и переменных. Сценарии же используют общее рабочее пространство. Как сценарии, так и функции не компилируются в машинный код и сохраняются в виде текстовых файлов.

Simulink – это компонент среды MATLAB, предназначенный для структурно-функционального имитационного моделирования и анализа динамических систем. Он состоит из инструмента составления блочных диаграмм и изменяемого набора библиотек блоков (элементов). Simulink позволяет провести интеграцию построенных диаграмм с остальной средой MATLAB. Возможен вызов на выполнение моделей Simulink из сценариев и функций MATLAB и вызов функций MATLAB из блоков Simulink. Компонент широко используется в области теории управления и цифровой обработки сигналов для моделирования и проектирования.

Решение задачи в системе WiseTasksDSP происходит на языке MATLAB. Принцип взаимодействия системы WiseTasksDSP и пакета MATLAB показан на рис. 2.

В пакете MATLAB выполняется одна виртуальная машина Java (Java Virtual Machine, JVM), а система WiseTasksDSP запущена на другой. Взаимодействие между ними происходит через интерфейс RMI (Remote Method Invocation, удаленный вызов методов) [9]. В состав MATLAB входит библиотека «jmi.jar», вызовы методов которой организуются при помощи свободно распространяемой библиотеки «MatlabControl.jar» [10]. Эта библиотека содержит «заглушки» (stubs) методов, реализуемых в «jmi.jar», и включена в систему WiseTasksDSP.

3. СТРУКТУРА СИСТЕМЫ WISETASKSDSP

Система состоит из модуля преподавателя и модуля студента. В модуле преподавателя педагог вводит условие задачи на предметно-ориентированном языке.

Задача по ЦОС имеет следующую структуру:

- Заголовок.
- Текстовое условие.
- Изображение к текстовому условию.
- Инструменты.
- Верификатор.

Интерес представляют последние две структурные единицы.

Инструменты – это список функций MATLAB (из стандартного набора) и элементов Simulink, доступных студенту при решении задачи. Также в него можно включать группы функций и элементов. Примеры: «функция fft» (для вычисления прямого ДПФ), «функция ifft» (для вычисления обратного ДПФ), «набор функций Signal Processing Toolbox» (библиотека функций для обработки сигналов), «элемент Сумма-

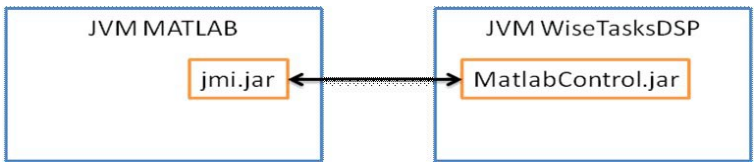

Рис. 2. Схема взаимодействия WiseTasksDSP и MATLAB

тор» (элемент Simulink, предназначенный для суммирования входов).

При решении задачи студенты также могут использовать любые арифметические функции и операции.

Верификатор – это логическая функция MATLAB, проверяющая решение ученика путем перебора значений параметров. Возвращает значение «истина», если решение признано верным на всех итерациях циклов. Особенностью функции является обязательное присутствие макрокоманды «Решение;», на место которой при проверке подставляется скрипт-решение студента.

Решение задачи происходит в модуле студента. Этот модуль позволяет загрузить задачу, ввести скрипт-решение, используя только разрешенные инструменты, и вызвать проверку, то есть запустить верификатор с подставленным в него скриптом-решением.

Упрощенная[1] грамматика разработанного предметно-ориентированного языка в расширенной форме Бэкуса-Наура [1] приведена в листинге 1 со следующими соглашениями:

::= – отделяет название лексемы от ее описания

{A} – ноль или более элементов A;

[A] – элемент A может входить или не входить;

(A B) – группа элементов;

A|B – либо A, либо B.

4. ПРИМЕР ЗАДАЧИ

Далее приведен пример структурированной задачи по ЦОС. Она посвящена нахождению верхней граничной частоты спектра аналогового (непрерывного) сигнала и проверки частоты дискретизации дискретного сигнала на допустимость для точного (без потерь информации) восстановления исходного (аналогового) сигнала из дискретного на основе теоремы Котельникова [11]. При определении верхней граничной частоты спектра аналогового сигнала используется 10% критерий (на практике также используются и другие критерии определения верхней граничной частоты), то есть за верхнюю граничную частоту принимается ближайшая к нулевой частота, при которой значение амплитудного спектра аналогового сигнала составляет 10% от максимума.

Условие задачи может быть сформулировано на предметно-ориентированном языке следующим образом:

Заголовок: Определение верхней граничной частоты спектра аналогового сигнала и проверка частоты дискретизации дискретного сигнала в соответствии с теоремой Котельникова.

Условие: Дан исходный аналоговый сигнал s(n) длины N (значения находятся в массиве S, T_an – шаг по времени для значений аналогового сигнала). Частота дискретизации дискретного сигнала Fs. С помощью

Листинг 1. Упрощенная грамматика предметно-ориентированного языка системы WiseTasksDSP

```
Описание ::= Заголовок ТекстовоеУсловие [Изображение] Инструменты Верификатор
Заголовок ::= "Заголовок: " Текст
ТекстовоеУсловие ::= "Условие: " Текст
Изображение ::= "Изображение к условию: " ИмяФайла
Инструменты ::= "Инструменты:" [{Инструмент}]
Верификатор ::= "Верификатор:" КодФункцииВерификатора
Инструмент ::= ТипИнструмента Идентификатор
ТипИнструмента::= "Функция" | "Элемент"|"Набор функций"|"Набор элементов"
Идентификатор ::= ('a'..'z'|'A'..'Z'|'_'){ ('a'..'z'|'A'..'Z'|'0'..'9'|'_')}
КодФункцииВерификатора ::= НачФрагментКода "Решение;" КонФрагментКода
```

[1] Упрощение грамматики состоит в отсутствии раскрытия нетерминалов «Текст», «ИмяФайла», «НачФрагментКода», «КонФрагментКода» ввиду неважности их формального описания для понимания структуры предметно-ориентированного языка.

MATLAB определить верхнюю граничную частоту спектра аналогового сигнала (Fup) по 10%-ному критерию и проверить, подходит ли данная частота дискретизации для представления данного сигнала в дискретном виде без потерь информации. Если подходит, присвоить переменной isOK значение true. Если не подходит, присвоить isOK=false и задать минимальную целую частоту дискретизации Fsnew, для которой выполняется условие теоремы Котельникова.

Инструменты: Набор функций Signal Processing Toolbox

Функция fft

Функция max

Верификатор см. листинг 2.

Листинг 2. Верификатор для задачи по цифровой обработке сигналов

```
function ver=ver_freqs()
   M=10;    % Количество тестов
   N=100;  % Количество отсчетов в массиве сигнала
   eps=0.001; % Константа для сравнения истинных частот и значений
   % частот, найденных студентом
   thresh=0.1; % Порог в соответствии с 10%-м критерием
   T_an=2.2e-5; % Шаг по времени для значений аналогового сигнала
   Fs=1000;  % Частота дискретизации, Гц
   flag=false; % Флаг правильности прохождения i-го теста
   for i = 1:M % Цикл тестов
          S=rand(N,1); % Массив случайных чисел с равномерным распределением
       % (массив значений аналогового сигнала)
     % Макрос для подстановки скрипта-решения студента
      Решение;
     % Проверка правильности решения
      ABS_S =abs(fft(S)); % Находим амплитудный спектр сигнала
      S_max=max(ABS_S); % Определяем максимальное значение амплитудного спектра
      S_up=thresh*S_max; % Определяем значение амплитудного спектра по
      % заданному критерию
      for j= 1:N % Ищем наименьшую частоту, на которой амплитудный спектр не
         if ABS_S(j)<=S_up % превышает порога
           Fup1=1/(T_an*N)*(j-1); % Истинная верхняя граничная частота спектра
           break;   % сигнала по заданному критерию
        end
      end
      if abs(Fup1-Fup) < eps % Сравниваем истинную граничную частоту с ответом
        % студента  (Fup - ответ студента)
         if Fs>=2*Fup1 % Проверяем выполнение условия теоремы Котельникова
            flag = isOK;
         else % Если оно не выполняется,
            Fsnew1 = ceil(2*Fup1); % то находим истинную наименьшую целую частоту
          % дискретизации, удовлетворяющую условию теоремы Котельникова
            flag = ~isOK & (abs(Fsnew1-Fsnew)<eps);
         end
      else
         flag=false;
      end
      if ~flag % Если тест пройден неправильно, то серия тестов прерывается
         break;
      end
   end
   ver=flag; % Если решение правильно прошло все тесты, то оно верно.
```

Задача, введенная в модуле преподавателя, приведена на рис. 3. Ввод условия осуществляется с помощью структурированного редактора предметно-ориентированных языков [4].

Для решения задачи необходимо ее загрузить в модуль студента. Данный модуль с загруженным примером задачи изображен на рис. 4. Окно модуля студента разделено на 3 части. В верхней части выводится текстовое условие задачи и изображение к нему (при его наличии). В средней части – панель инструментов, в которой отражены заданные в условии инструменты. И, наконец, в нижней части находится поле для ввода скрипта-решения на языке MATLAB. Для упрощения использования в скрипте функций из набора инструментов можно воспользоваться панелью инструментов. Чтобы ввести функцию, не принадлежащую ни к одному набору (toolbox), необходимо щелкнуть мышью по имени функции на панели инструментов. Если требуется воспользоваться мо щелкнуть мышью под именем этого набора и выбрать нужную функцию из раскрываю-

щегося списка. И панель инструментов, и поле ввода скрипта-решения реализованы на основе структурированного редактора.

Для проверки решения требуется нажать на среднюю кнопку в панели инструментов окна (под верхнем меню). После проверки на экран будет выведено сообщение о правильности или неправильности ответа студента.

Скрипт-решение для рассматриваемой задачи представлен в листинге 3.

В ходе проверки приведенная в листинге 2 функция-верификатор вначале определяет начальные условия для выполнения тестов. Далее в цикле организуется тестирование решения студента. На каждом тесте выполняются следующие шаги:

1. Определение начальных условий для выполнения решения.
2. Вызов скрипта-решения.
3. Проверка правильности ответов, определенных в п. 2.

Рассчитанные ответы зависят от массива отсчетов сигнала, задаваемого на каждом тесте случайным образом, поэтому студен-

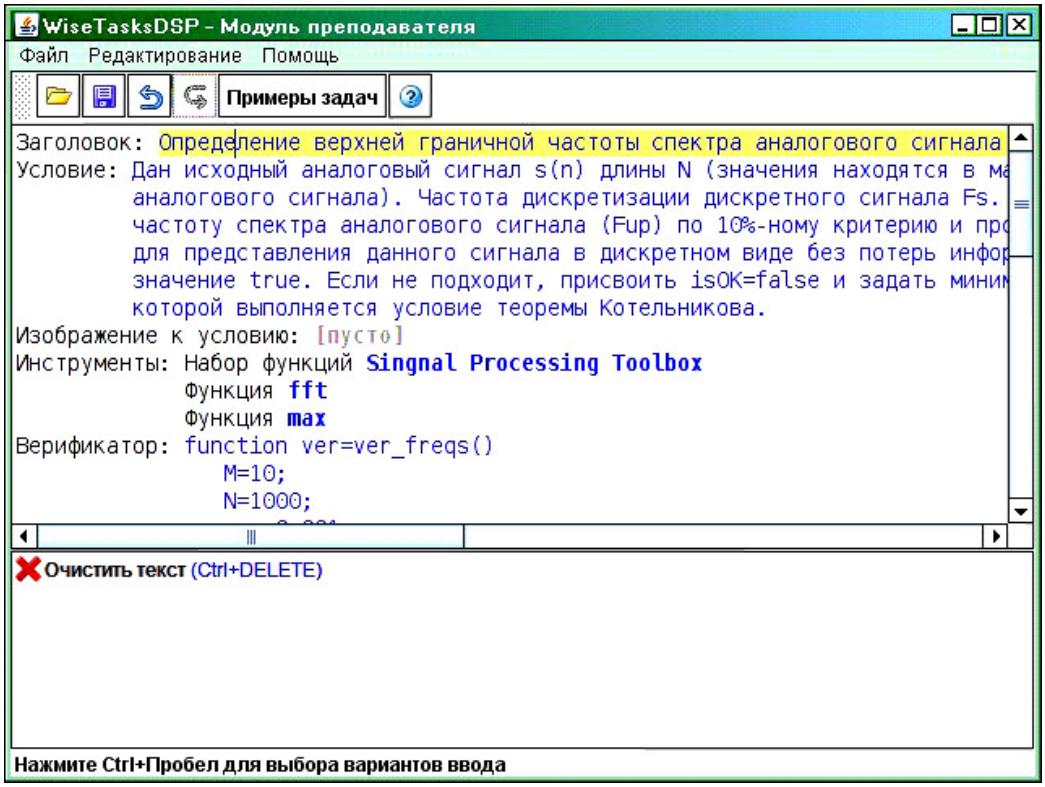

Рис. 3. Модуль преподавателя системы WiseTasksDSP

Листинг 3. Скрипт-решение примера задачи

```
ABS_S=abs(fft(S)); % Находим амплитудный спектр сигнала
S_max=max(ABS_S); % Определяем максимальное значение амплитудного спектра
S_up=0.1*S_max; % Определяем значение амплитудного спектра по заданному критерию
for j= 1:N % Ищем наименьшую частоту, на которой амплитудный спектр не
   if ABS_S(j)<=S_up % превышает порога
      Fup=1/(T_an*N)*(j-1); % Верхняя граничная частота
      break;
   end
end
if Fs>=2*Fup % Проверяем выполнение условия теоремы Котельникова
   isOK=true;
else  % Если оно не выполняется,
   Fsnew = ceil(2*Fup); % то находим наименьшую целую частоту
           % дискретизации, удовлетворяющую условию теоремы Котельникова
   isOK = false;
end
```

ту бессмысленно сводить скрипт-решение к простому присваиванию переменным Fs и Fup числовых значений. В этом случае ответы будут неверны наверняка.

5. ФОРМАТ ХРАНЕНИЯ ОПИСАНИЯ ЗАДАЧИ

Условие задачи хранится в формате XML. Описание задачи находится внутри тега «task», который имеет атрибут «name» – имя задачи. Оно состоит из списка инструментов, верификатора и текстового условия задачи.

Инструменты перечисляются внутри тега «tools». Каждый инструмент представляет собой тег «tool» с атрибутами «type» – тип (функция MATLAB, набор функций (toolbox), элемент Simulink, набор элементов (blockset)) и «name» – название инстру-

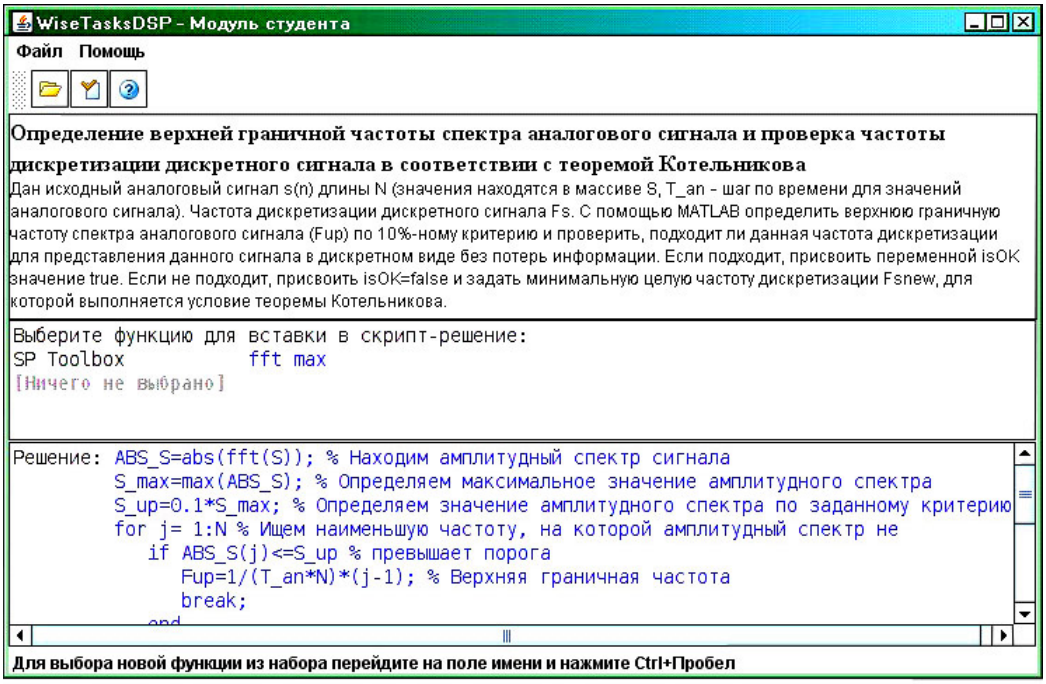

Рис. 4. Модуль студента с загруженным примером задачи

Листинг 4. Пример XML-файла с задачей (с сокращениями)

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<task title=" Определение верхней граничной частоты …">
  <picture />
<description>
<![CDATA[
Дан исходный аналоговый сигнал s(n) длины N ……
  ]]>
  </description>
<tools>
  <tool name="SP_TOOLBOX" type="Toolbox" />
  <tool name="FFT" type="Func" />
 <tool name="MAX" type="Func" />
 </tools>
 <verifier>
<![CDATA[
function ver=ver_freqs()
……
]]>
  </verifier>
  </task>
```

мента. Текст функции-верификатора содержится внутри тега «verifier», а текстовое условие задачи – в теге «description». Изображение к условию прикрепляется с помощью тега «picture» (если изображения нет, указывается пустой тег). Пример XML-файла с задачей приведен в листинге 4.

ЗАКЛЮЧЕНИЕ

В данной статье представлена модель самопроверяемой задачи и показаны возможности одного из приложения данной модели – системы WiseTasksDSP – для решения учебных задач. В качестве примера продемонстрирована одна из часто встречающихся учебных задач по ЦОС, связанная с определением верхней граничной частоты спектра аналогового сигнала и проверкой частоты дискретизации дискретного в соответствии с теоремой Котельникова. Система WiseTasksDSP может также использоваться для решения ряда других известных в ЦОС задач. К их числу относятся, например, поиск оптимальных по методу наименьших квадратов коэффициентов линейного предсказания [11–14], синтез КИХ-фильтров с АЧХ, наименее отличающейся от эталонной в смысле заданного критерия [11–14], синтез адаптивных фильтров [11–14], построение оптимального классификатора в двумерном пространстве [15] и пр. Система WiseTasksDSP также является перспективной для многих научных задач, актуальных для практики, сложность которых на несколько порядков выше, по сравнению с теми задачами, что используются в учебных целях.

Следующий шаг в разработке системы – введение строгого ограничения на применяемые в решении студента функции (на данном этапе ограничения носят рекомендательный характер). Реализация будет осуществляться с помощью функции MATLAB *depfun*, которая, в частности, возвращает список всех используемых в M-файле функций [8].

Литература

1. *Фаулер М.* Предметно-ориентированные языки программирования. М.–СПб.–Киев: Вильямс, 2011.

2. *Корнеев Г.А., Станкевич А.С.* Методы тестирования решений задач на соревнованиях программированию // Труды II межвузовской конференции молодых ученых. СПб.: СПбГУ ИТМО, 2005. С. 36–40.

3. *Богданов М.С.* Автоматизация проверки решения задач по формальному описанию ее условия // Компьютерные инструменты в образовании, 2006. № 4. С. 51–57.

4. *Перченок О.В., Поздняков С.Н., Посов И.А.* Автоматизация проверки решения геометрических задач по описанию их условий на предметно-ориентированном языке // Компьютерные инструменты в образовании, 2012. № 1. С. 37–44.

5. *Перченок О.В.* Разработка системы электронного обучения для автоматической верификации решений задач по описанию условия на предметно-ориентированных языках. ООО «Технолит», Материалы XVIII Международной научно-методической конференции «Современное образование: содержание, технологии, качество». СПб., 2012. Т. 1. С. 223–224.

6. Сайт системы динамической геометрии GeoGebra: http://www.geogebra.org (дата обращения: 10.06.2012).

7. *Леоненков А.В.* Самоучитель UML 2. СПб.: БХВ-Петербург, 2007.

8. *Дьяконов В.П.* MATLAB 7.*/R2006/2007. Самоучитель. М.: «ДМК-Пресс», 2008.

9. *Хорстманн К., Корнелл Г.* Java 2. Библиотека профессионала. Том 2. Тонкости программирования. М.–СПб.–Киев: Вильямс, 2010.

10. Описание библиотеки MatlabControl: http://undocumentedmatlab.com/blog/jmi-wrapper-remote-matlabcontrol/ (дата обращения: 10.06.2012).

11. *Сергиенко А.Б.* Цифровая обработка сигналов. 3-е издание. СПб.: БХВ-Петербург, 2011.

12. *Солонина А.И., Арбузов С.М.* Цифровая обработка сигналов. Моделирование в MATLAB. СПб.: БХВ-Петербург, 2008.

13. *Айфичер Э., Джервис Б.* Цифровая обработка сигналов. М.–СПб.–Киев: Вильямс, 2004.

14. *Солонина А.И., Улахович Д.А., Арбузов С.М., Соловьева Е.Б.* Основы цифровой обработки сигналов. 2-е изд. СПб.: БХВ-Петербург, 2005.

15. *Айвазян С.А., Мхитарян В.С.* Прикладная статистика в задачах и упражнениях: Учебник для вузов. М.: ЮНИТИ-ДАНА, 2001.

Abstract

The present paper discusses a new approach to task verification automation in the field of digital signal processing. The suggested approach employs a special domain-specific language of the WiseTasksDSP system, which has been developed by the authors. The example illustrating the effectiveness of the approach is concerned with finding the upper frequency limit of a continuous signal and the sampling rate of a discrete signal for the purpose of reconstructing the continuous signal from a discrete signal without information loss (Kotelnikov theorem). Furthermore, other problems are mentioned, where the application of WiseTasksDSP might prove to be useful.

Keywords: domain-specific language, digital signal processing, MATLAB, Java.

Перчёнок Олег Владимирович,
аспирант СПбГЭТУ «ЛЭТИ»,
факультет Компьютерных технологий и информатики (ФКТИ),
кафедра ВМ-2,
olegperch@gmail.com

Клионский Дмитрий Михайлович,
аспирант СПбГЭТУ «ЛЭТИ»,
факультет Компьютерных технологий и информатики (ФКТИ),
кафедра Математического обеспечения и применения ЭВМ (МОЭВМ),
klio2003@list.ru