Применение методов творческого мышления и ТРИЗ при разработке приложений для Intel Atom.

Основы ТРИЗ.

4 день, 29.01.2011

Зимняя Школа на базе
Лаборатории СПРИНТ
(СПбГУ-Intel)

Санкт-Петербург

Преподаватели и разработчики курса

Рубин Михаил Семенович

- Мастер ТРИЗ, ученик и соавтор Г.С. Альтшуллера
- Директор, а затем Президент Международной Ассоциации ТРИЗ в 1997 г. - 2005 г.г.
- Ученый секретарь диссертационного совета МА ТРИЗ с 2006 г.
- Специалист в области развития малого бизнеса
- Директор по маркетингу, ведущий научный сотрудник ЦИТК "Алгоритм"
- Провел более 90 семинаров по ТРИЗ, автор более 100 статей по ТРИЗ

Одинцов Игорь Олегович

- Менеджер по стратегическому развитию Intel в РФ
- Старший преподаватель СПбГУ, автор учебника по программированию
- Автор статей по применению ТРИЗ в программировании

Герасимов Олег Михайлович

- К.т.н., Доцент, Мастер ТРИЗ
- Начальник отдела обучения ЦИТК "Алгоритм"
- участвовал в 33 проектах, из них в 11 в качестве ГИП
- Провел более 30 семинаров по обучению методике G3-ID.
- Автор более 20 публикаций по методике G3-ID и ее применению.

Преподаватели и разработчики курса

Сысоев Сергей Сергеевич, к.ф.-м.н.

Специалист по ТРИЗ, 3-й уровень.

Приемы разрешения технических противоречий в применении к задачам ИТ

Зиненко Ольга Игоревна,

Выпускница матмех СПбГУ, 2010 год

. Систематизация и анализ паттернов проектирования на основе стандартов ТРИЗ.

Струсь Глеб Игоревич,

Выпускник матмех СПбГУ, 2010 год.

Постановка и решение изобретательских задач в программировании на основе методов ТРИЗ.

Примерное расписание на 26-30 января 2011

26 января

- Введение в курс ТРИЗ
- Изобрета-тельские задачи
- Противоречия требований
- Приемы решения противоречий
- Таблица применения приемов
- Учебные задачи

27 января

- Идеальность и ресурсы
- Законы развития техники
- Введение в прогнозиро-вание
- Противоречия свойств. Принципы решения противоречий
- Введение в РТВ
- Элеполи и стандарты
- Консультации

28 января

- Функциональ-ный анализ
- Свертывание
- Оперативная зона
- Системный оператор
- Введение в АРИЗ
- Учебные задачи
- Консультации

29 января

- Постановка и решение изобретатель-ских задач
- Свертывание в программах
- АРИЗ-Универсал-2010
- Учебные задачи
- Консультации

30 января

- ТРИЗ и бизнес
- Элементы прогнози-рования
- Диагностика в ТРИЗ

Постановка задач и свертывание при развитии программных продуктов

Постановка задачи и свертывание в программировании

- **Цель:** обоснование применимости методов ТРИЗ для решения изобретательских задач в программировании и создание программного продукта, основанного на алгоритмах постановки изобретательских задач
- **Задачи:**
  - Исследовать Алгоритм Решения Изобретательских Задач (АРИЗ)
  - Обосновать применимость АРИЗ для решения изоретательских задач в программировании
  - Исследовать алгоритмы постановки изобретательских задач
  - Определить методы анализа систем, необходимые для осуществления алгоритмов постановки
  - Адаптировать выбранные алгоритмы и методы для программной реализации
  - Создать программный продукт, основанный на выбранных алгоритмах

Обоснование

Успешное решение ряда задач из области программирования убедительно доказывает возможность эффективного применения методов ТРИЗ для решения изобретательских задач в программировании.

Задача об ускорении сортировки массива

Постановка

Массив, содержащий целые числа, можно отсортировать методом пузырька. Время выполнения алгоритма растет квадратично длине массива. Как ускорить сортировку, используя существующий алгоритм сортировки?

Задача о многорежимности в графическом редакторе

Постановка

В редакторе геометрических фигур существует два режима редактирования, между которыми приходится часто переключаться. Первый режим позволяет передвигать вершины фигуры. Второй – добавлять новые вершины путем разбиения ребра на две части. Как сделать использование этих двух режимов удобным?

Алгоритм проведения свертывания

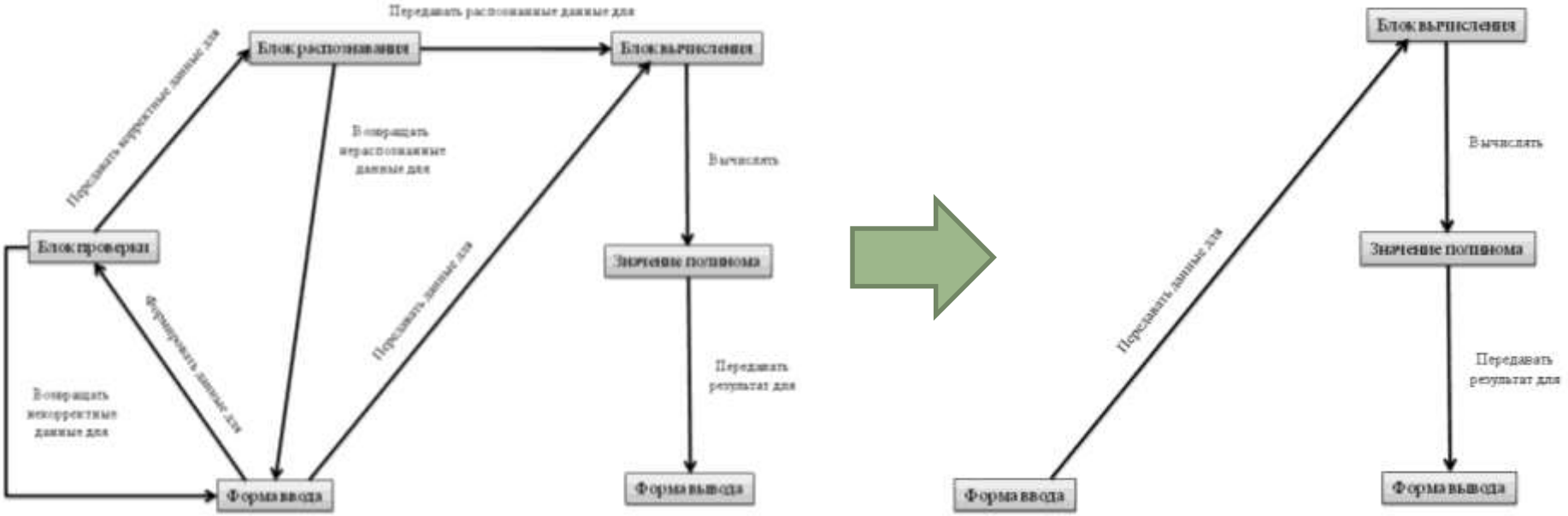
- Вспомогательные алгоритмы
  - Построение компонентной модели
  - Построение структурной модели
  - Построение функциональной модели
  - Построение причинно-следственных цепочек
- Алгоритм свертывания

Задача о полиноме.

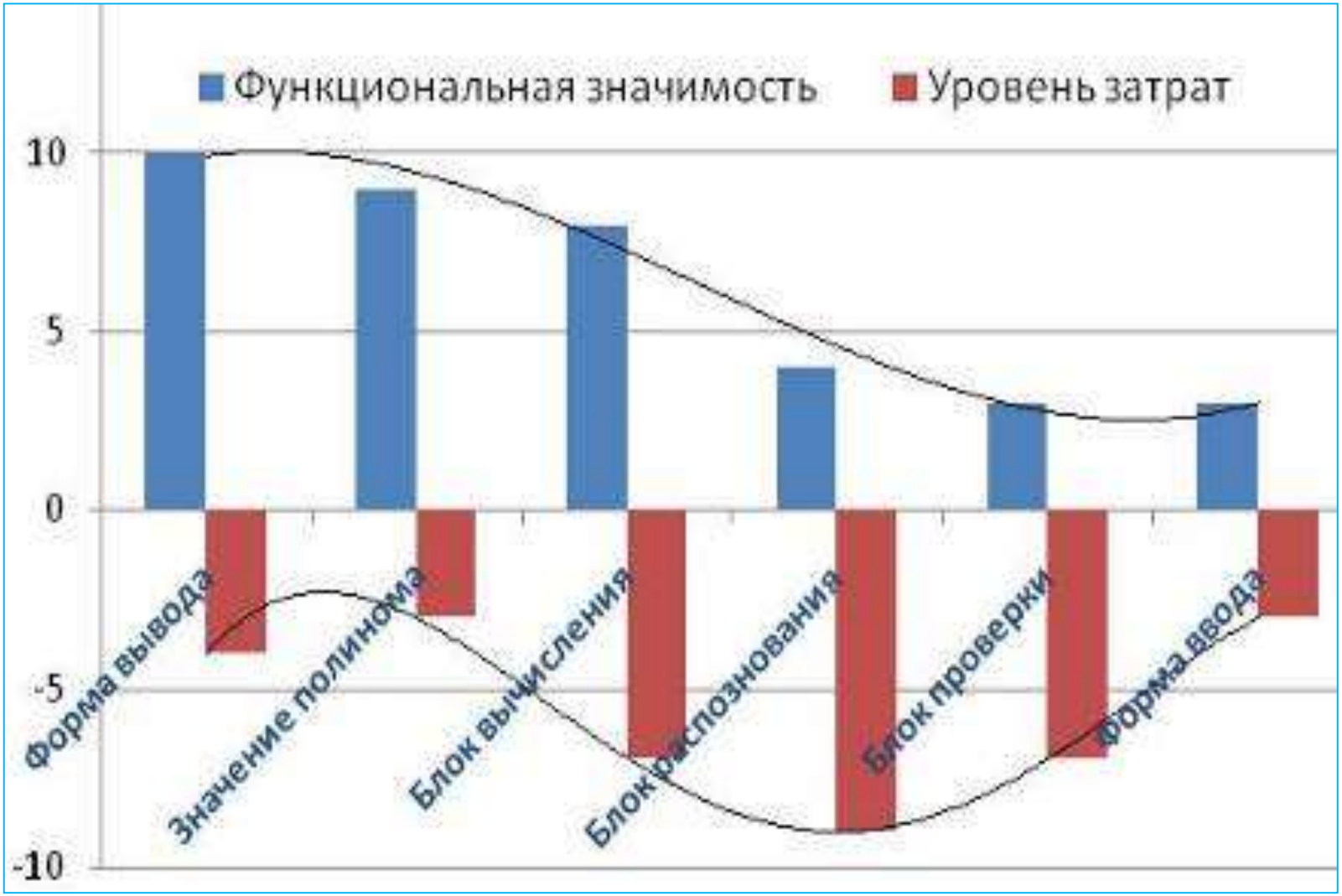
- **Задача 7.** Задача о программе вычисления произвольного полинома. В программе, предназначенной для вычисления значения произвольного полинома, имеется текстовое поле для ввода. Введенная строка затем проверяется на правильность (действительно ли пользователь ввел полином, а не произвольную строку), а затем распознается (создается модель полинома для его последующего вычисления). Однако, написание такой программы вызывает трудности, возникает большое количество ошибок. Кроме того, усложняется структура выполнения программы. Необходимо упростить и повысить надежность программы

Применение свертывания для задач из программирования

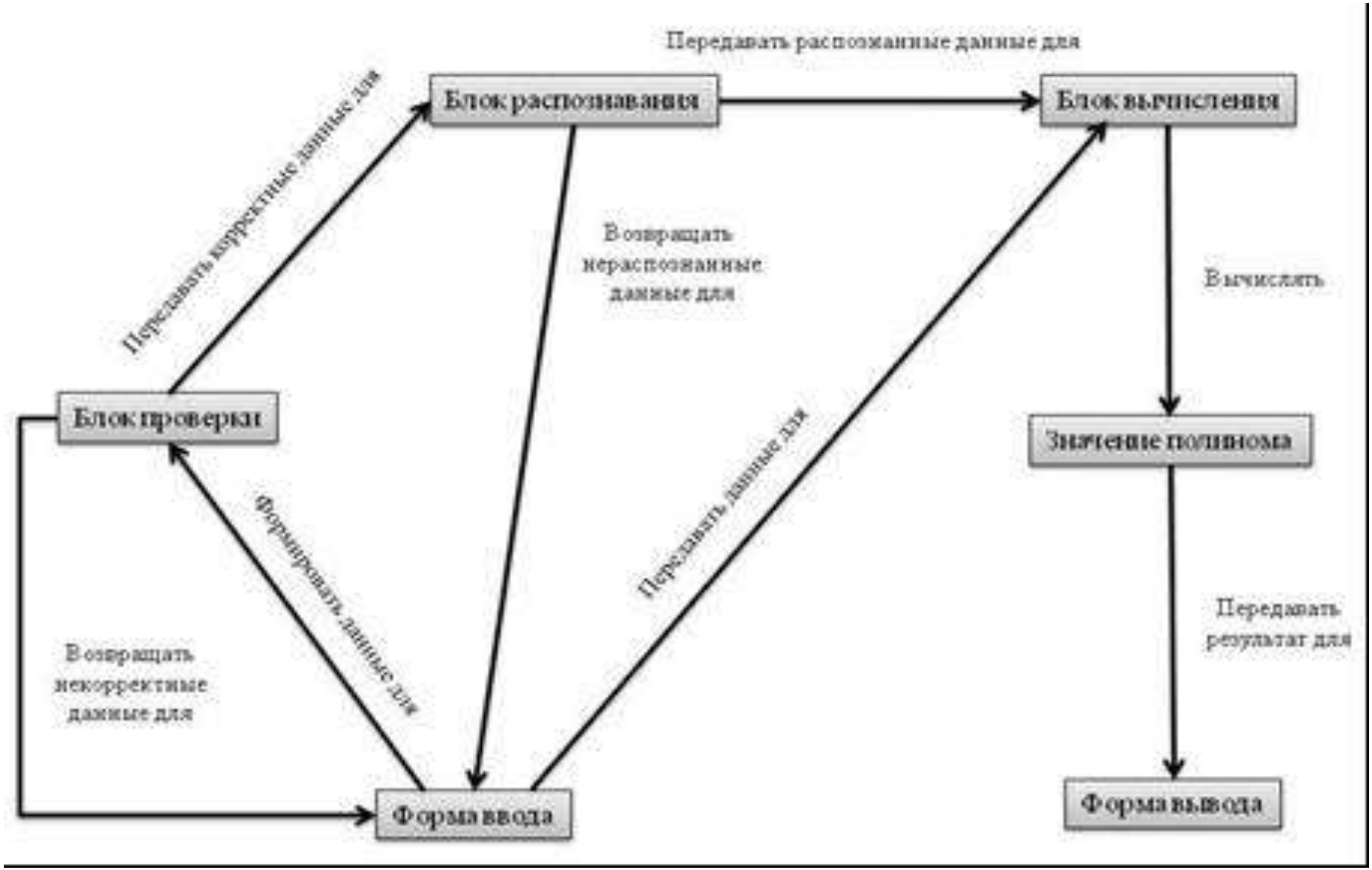
На примере программы, вычисляющей значение полинома, показана процедура свертывания с выходом на постановку изобретательской задачи и ее дальнейшим решением



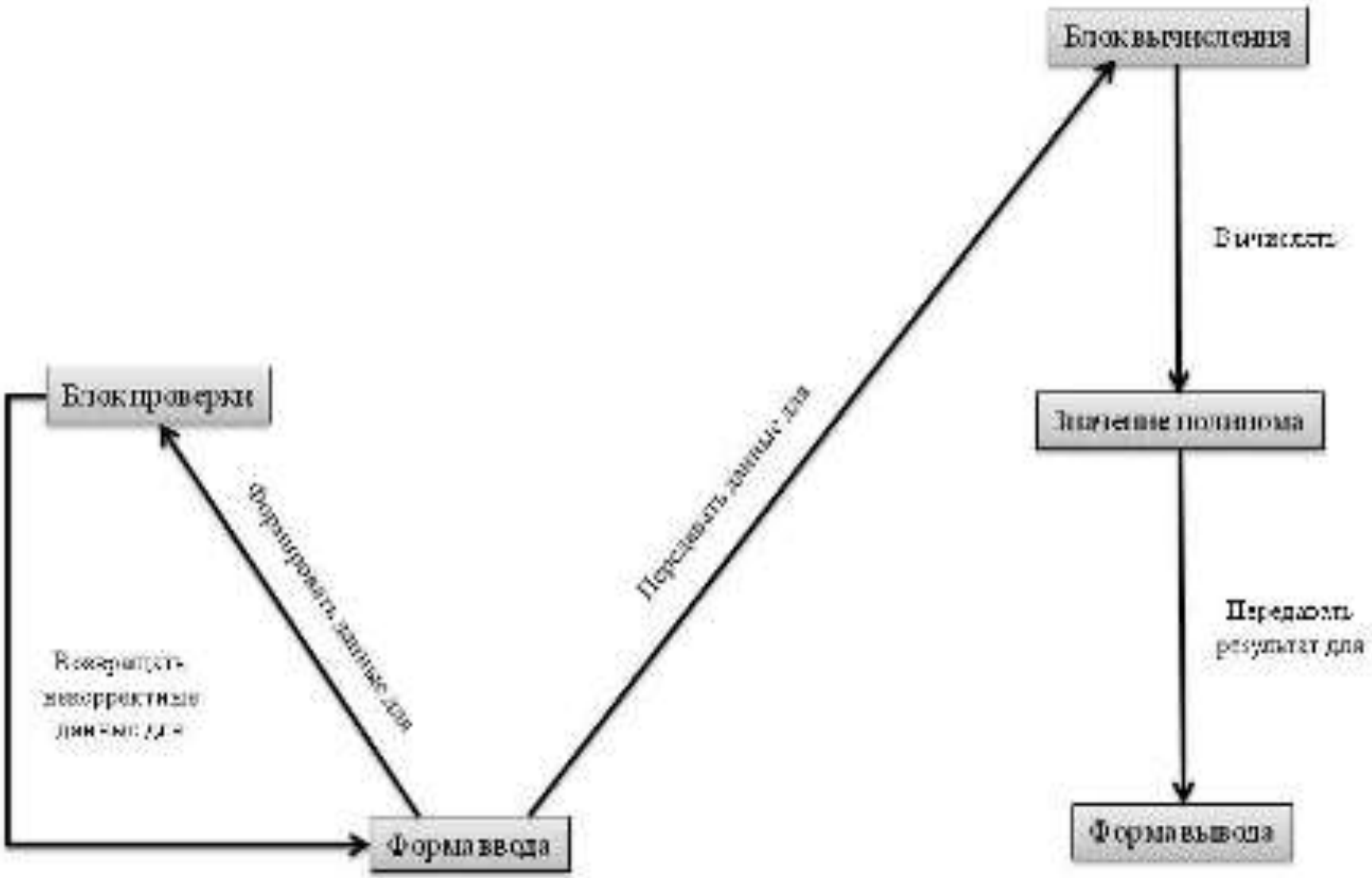
Анализ функциональной значимости элементов



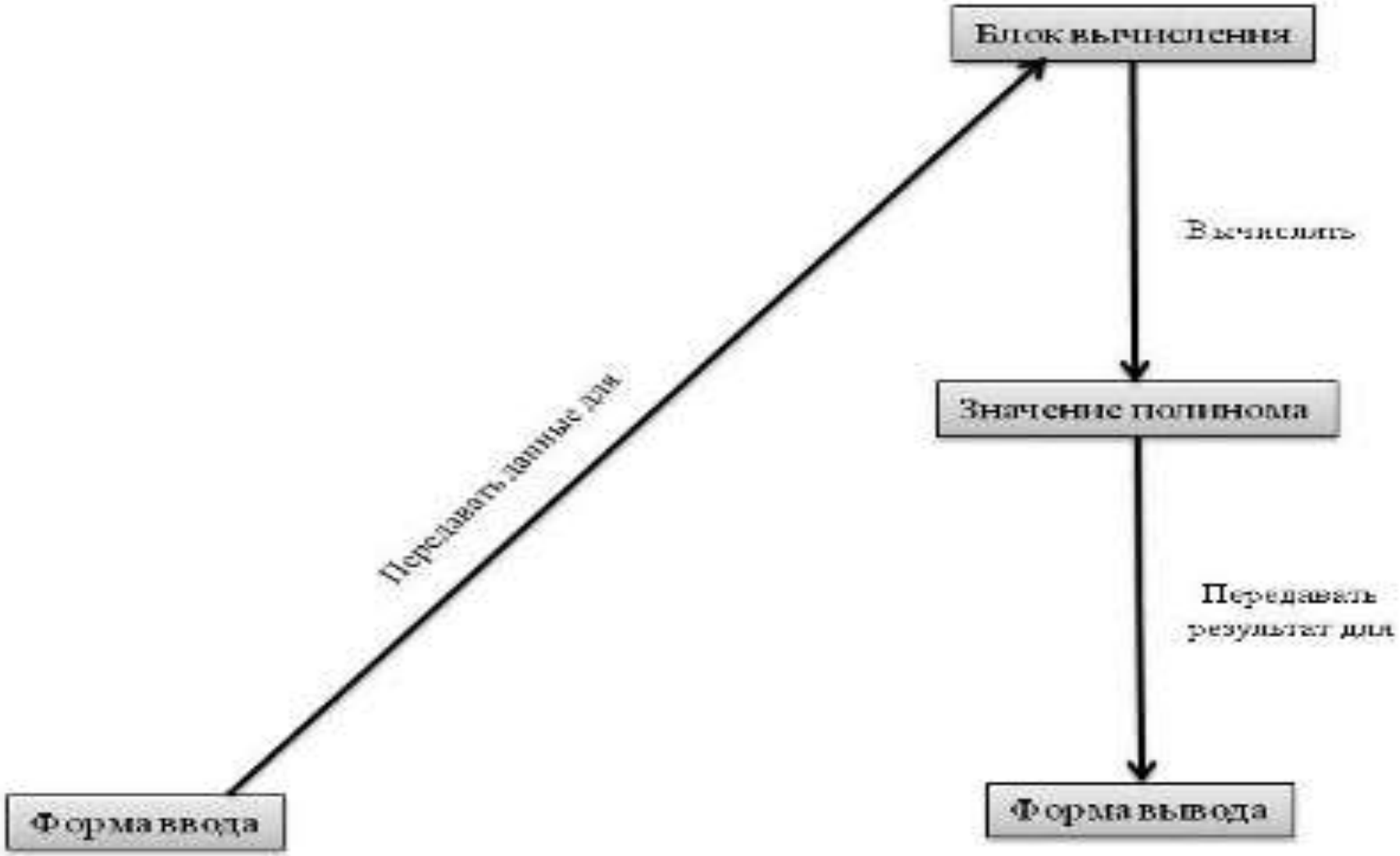
Исходная модель



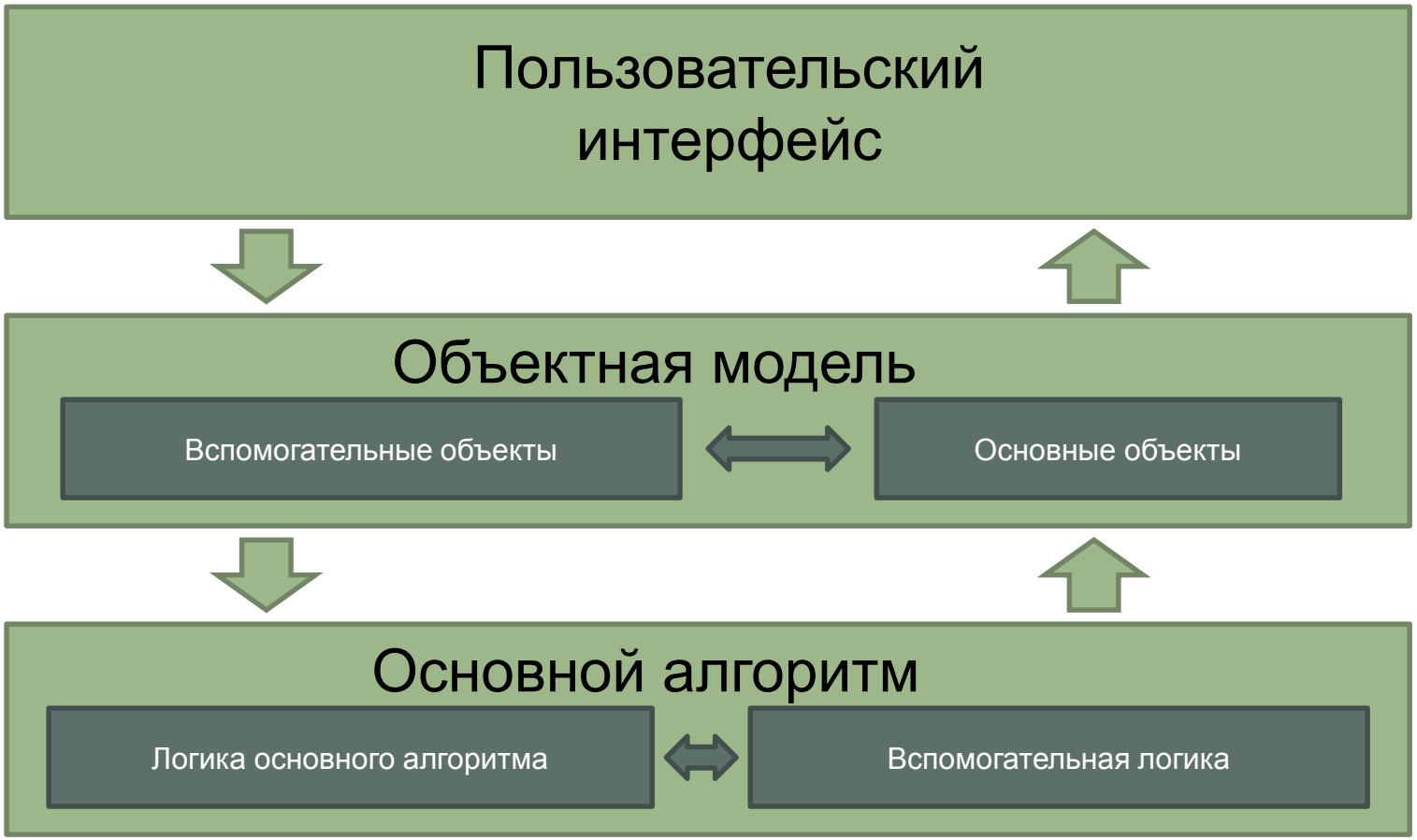
Свертывание блока распознавания



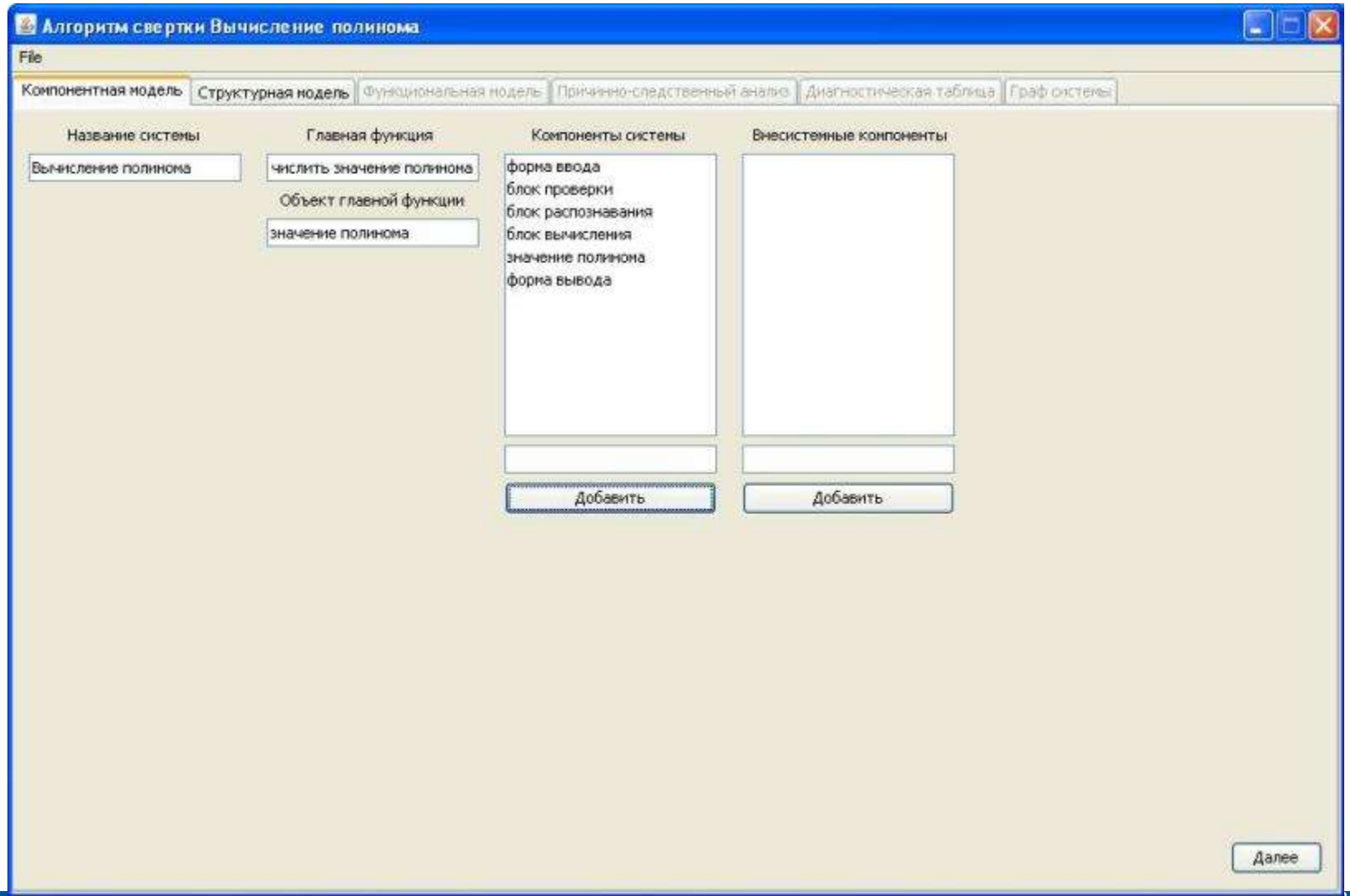
Свертывание блока проверки



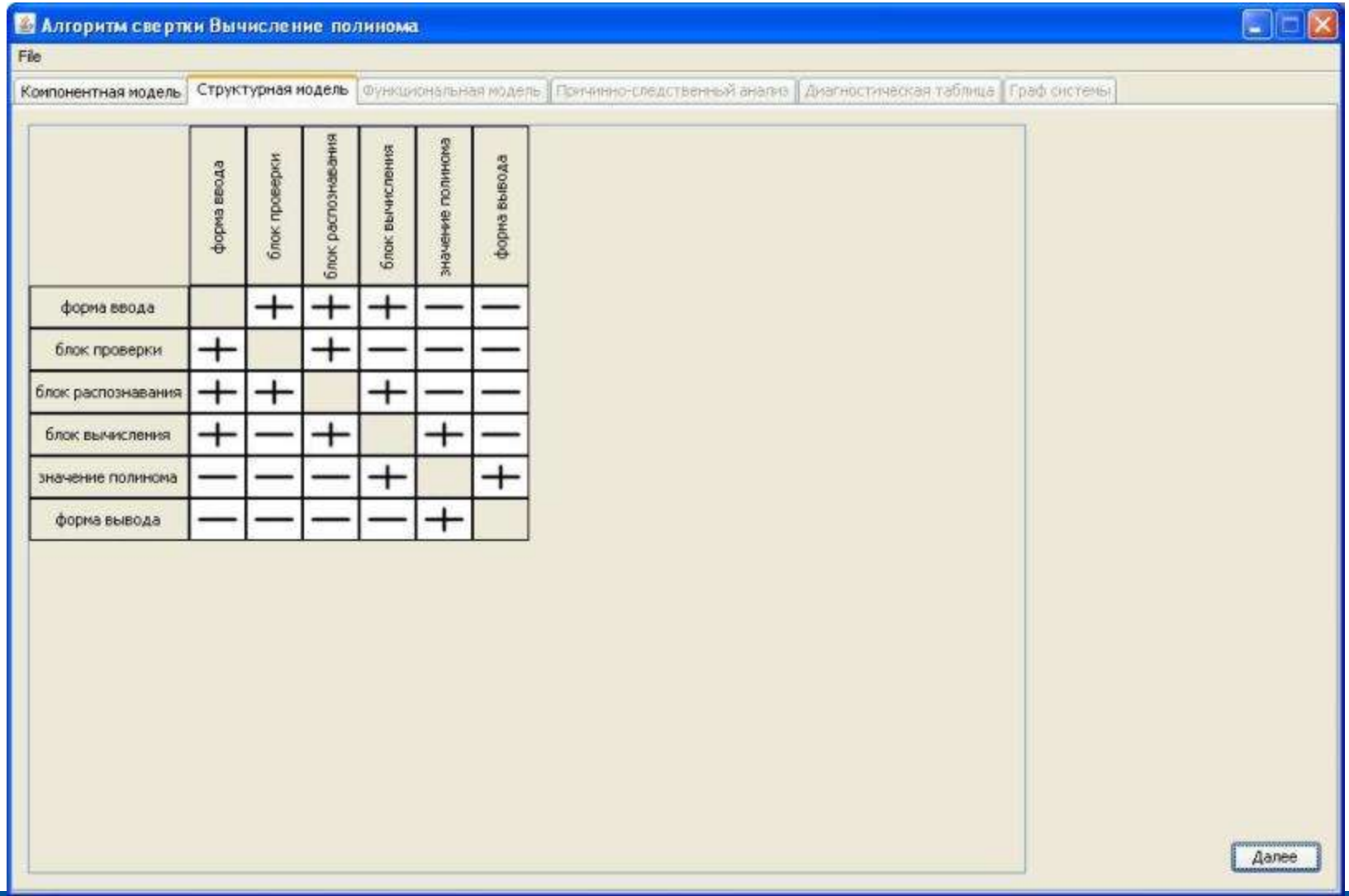
Архитектура разработанной системы



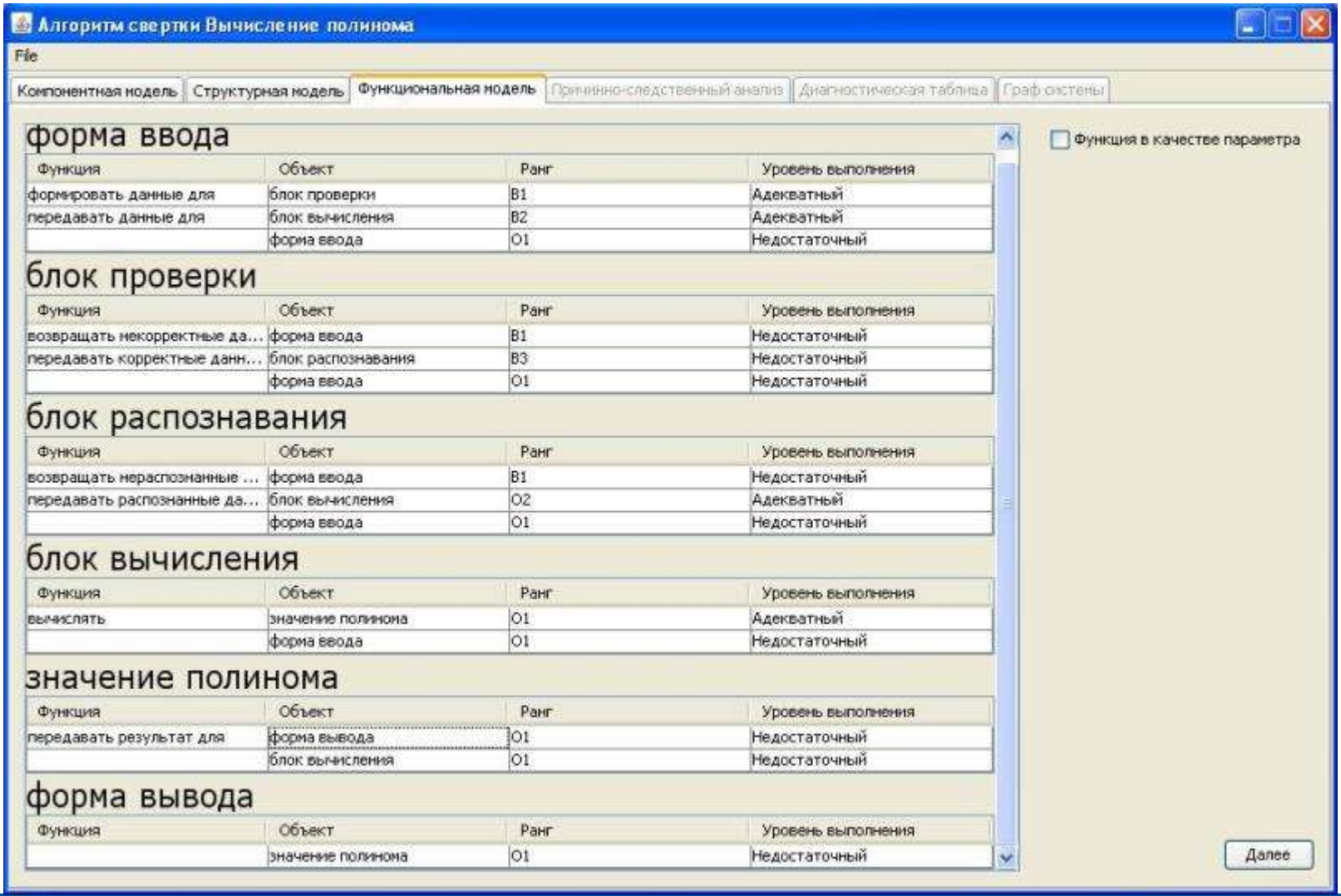
Работа с пользовательским интерфейсом системы



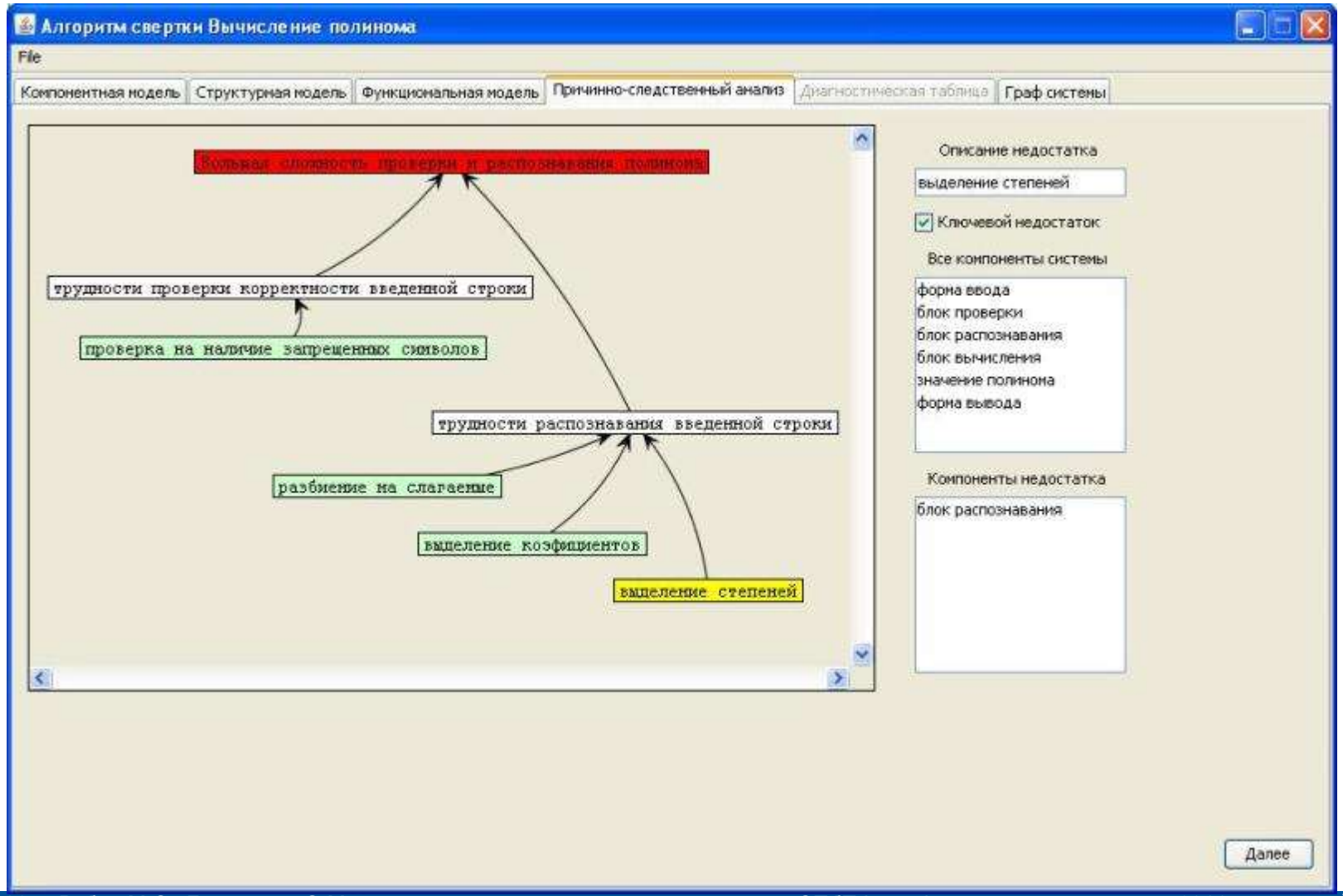
Работа с пользовательским интерфейсом системы



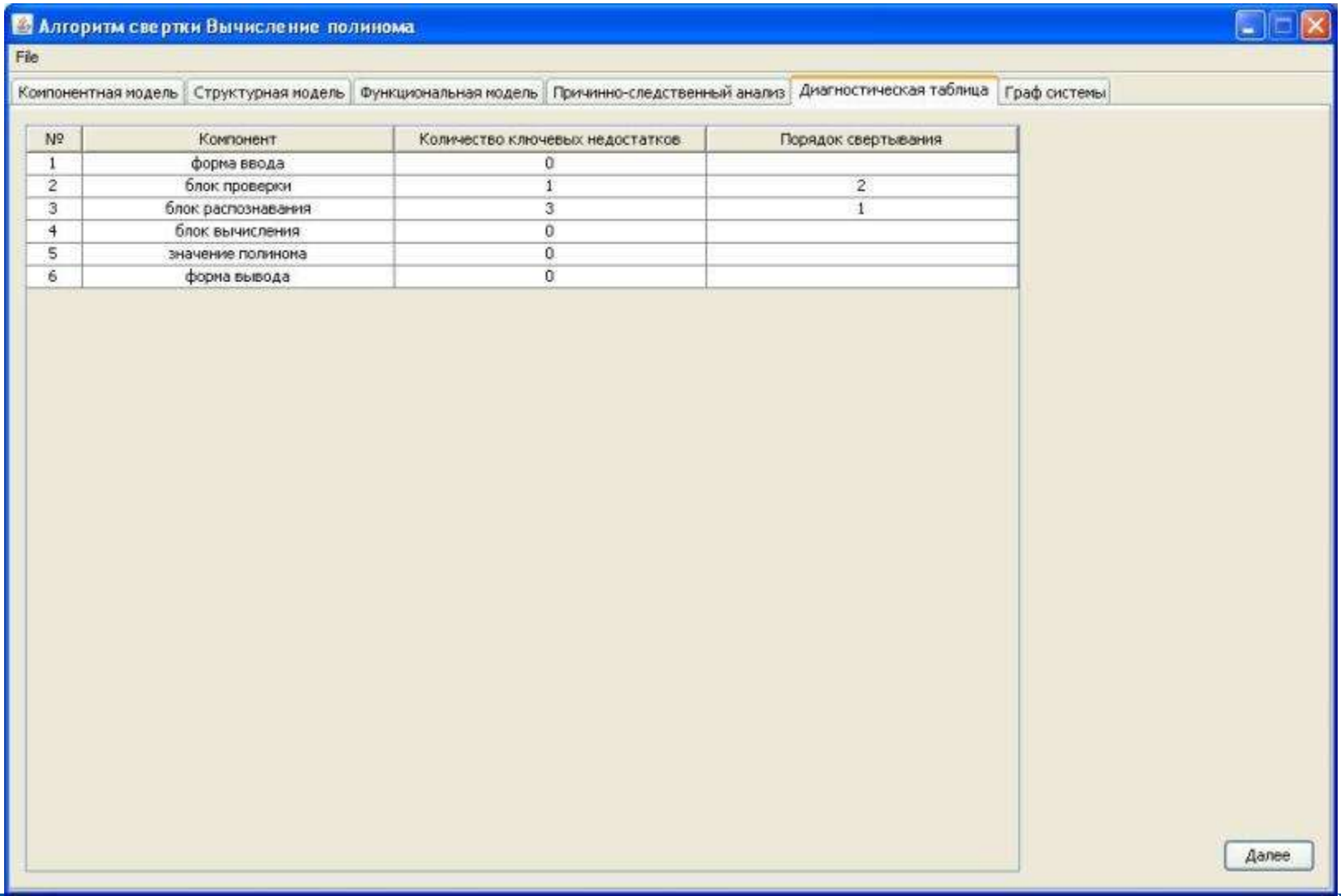
Работа с пользовательским интерфейсом системы



Причинно-следственный анализ

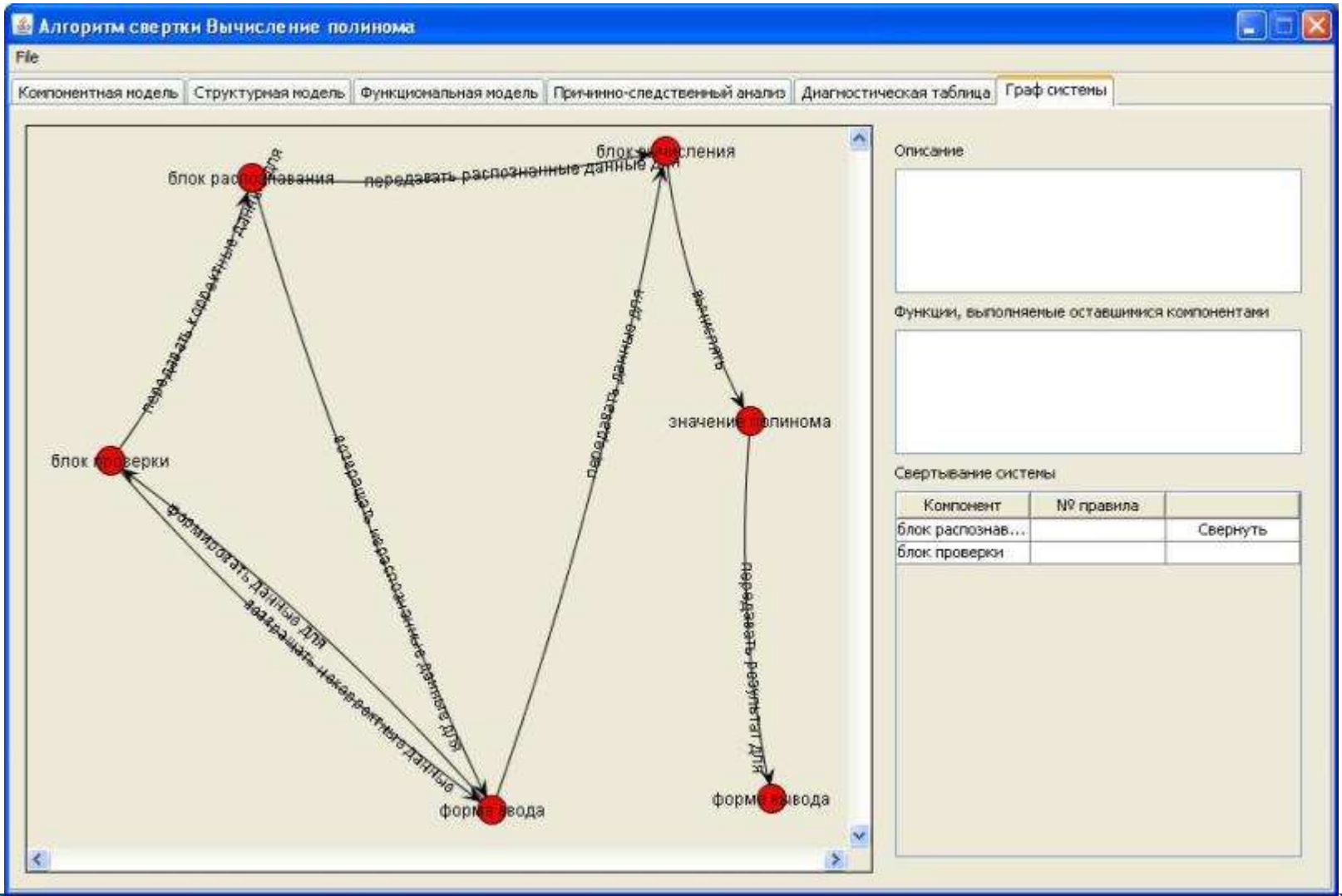


Диагностическая таблица



№	Компонент	Количество ключевых недостатков	Порядок свертывания
1	форма ввода	0	
2	блок проверки	1	2
3	блок распознавания	3	1
4	блок вычисления	0	
5	значение полинома	0	
6	форма вывода	0	

Граф системы



АРИЗ-Универсал-2010

Задача о сортировке массива.

- **Задача 6.** Задача о сортировке массива. Массив, например содержащий целые числа, можно отсортировать методом пузырька. Однако время выполнения алгоритма растет квадратично длине массива, то есть для достаточно длинных массивов это время становится недопустимо большим. Можно создать новый алгоритм сортировки, но на это также потребуется много времени. Создание нового метода “с нуля” является трудной задачей. Как сократить время сортировки длинного массива, не создавая новый алгоритм сортировки?

Решение задачи по АРИЗ-Универсал-2010

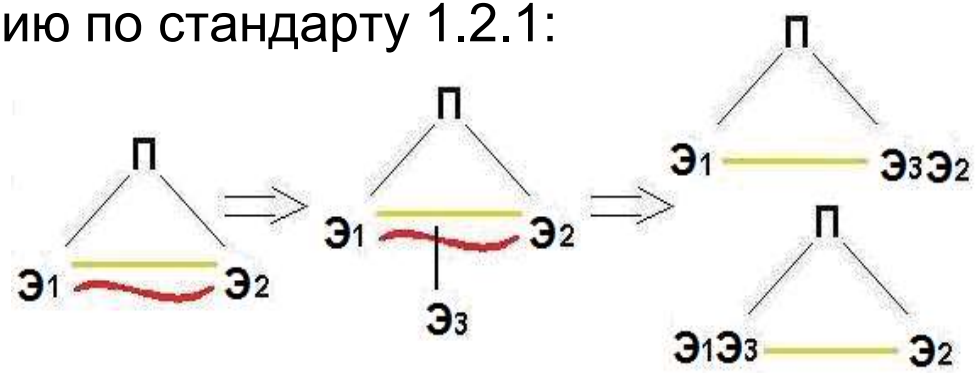
- **Функциональный ИКР**: Массив САМ уменьшает время сортировки во время работы алгоритма при использовании существующего алгоритма сортировки.
- **Противоречия требований**: ЕСЛИ создается новый алгоритм, ТО уменьшается время сортировки массива, НО на создание нового алгоритма уйдет слишком много времени.
- ЕСЛИ используется существующий алгоритм, ТО массив сортируется, НО сортировка занимает слишком много времени.
- **Использование сокращенной таблицы применения приемов** устранения противоречий. Можно использовать две пары требований:
- Скорость и Удобство изготовления (9-32)
- Скорость и Сложность устройства (9-36)
- Получаем рекомендации по приемам: 13, 1, 10, 4, 34 (дробления, предварительного действия, асимметрии, отброса и регенерации частей).

Решение задачи по АРИЗ-Универсал-2010

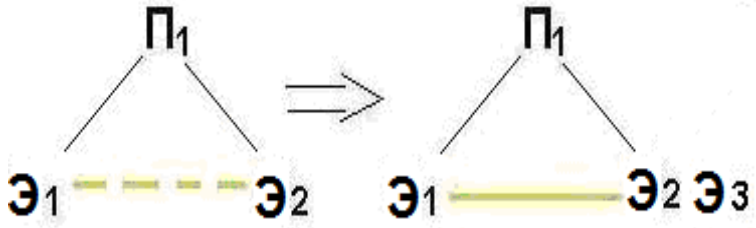
- Получаем **рекомендации по приемам**: 13, 1, 10, 4, 34 (дробления, предварительного действия, асимметрии, отброса и регенерации частей).
- **Конфликтующие элементы**: Массив целых чисел (в двух состояниях – отсортированные и не отсортированы), Алгоритм сортировки. Они взаимодействуют между собой, можно ограниченно менять массив целых чисел.
- **ОВ** – время сортировки массива
- **ОЗ** – взаимодействие алгоритма сортировки с массивом целых чисел
- **Внутрисистемные ресурсы**: ресурсы массива целых чисел.

Решение задачи по АРИЗ-Универсал-2010

- **Элепольное решение:** Э1 – неотсортированный массив, Э2 – отсортированный массив, П – алгоритм сортировки. Так как размер массива отрицательно влияет на время работы алгоритма, то между Э1 и Э2 есть одновременно и полезная связь (массив все же сортируется) и вредная – это происходит слишком долго. Получаем рекомендацию по стандарту 1.2.1:

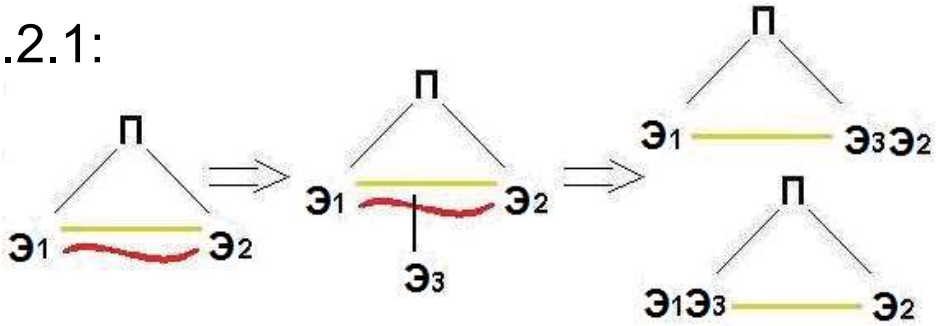


- Либо взаимодействие между Э1 и Э2 можно считать не достаточно эффективным и тогда нужно использовать рекомендации стандарта 2.1:

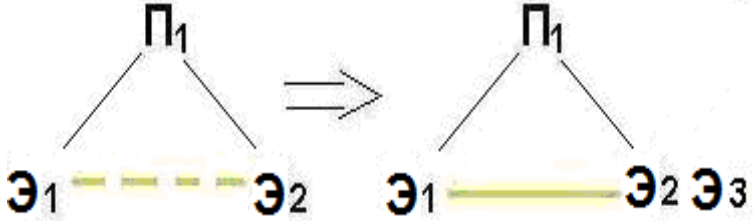


Решение задачи по АРИЗ-Универсал-2010

- **Элепольное решение:** Э1 – неотсортированный массив, Э2 – отсортированный массив, П – алгоритм сортировки. Так как размер массива отрицательно влияет на время работы алгоритма, то между Э1 и Э2 есть одновременно и полезная связь (массив все же сортируется) и вредная – это происходит слишком долго. Получаем рекомендацию по стандарту 1.2.1:



- Либо взаимодействие между Э1 и Э2 можно считать не достаточно эффективным и тогда нужно использовать рекомендации стандарта 2.1:



- Линия введения элементов предлагается использовать для введения уже имеющиеся элементы или их модификации.

Решение задачи по АРИЗ-Универсал-2010

- **Ресурсный ИКР.** Икс-элемент из ресурсов системы САМ УСТРАНЯЕТ большое время сортировки, СОХРАНЯЯ использование существующего алгоритма сортировки.
- **Противоречие свойств**. Конфликтующий элемент (массив чисел) должен быть малого размера, чтобы обеспечить малое время выполнения алгоритма сортировки, и должен быть большим, чтобы хранить все необходимые числа.
- **ИКР свойств**. Взаимодействие алгоритма сортировки с массивом целых чисел в течение сортировки должно САМО обеспечивать малый размер обрабатываемого массива и большой размер получаемого массива.

Решение задачи по АРИЗ-Универсал-2010

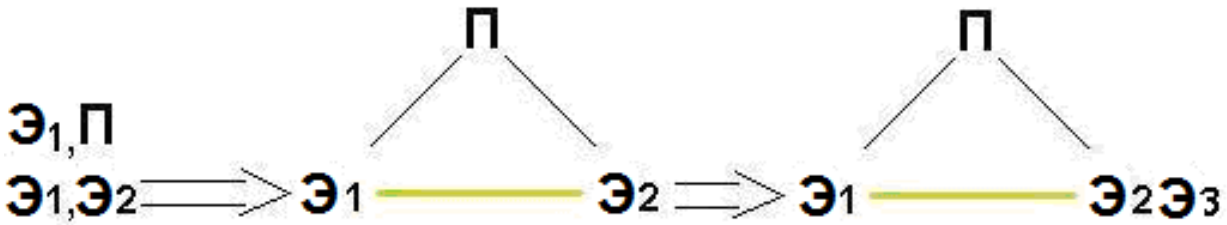
- **Решение.** Согласно принципу дробления и принципу предварительного действия, нужно до начала сортировки разбить массив на части (например, на две равные части). При этом каждый из двух новых массивов будут меньше начального. Затем нужно применить алгоритм сортировки к каждому массиву по отдельности, после чего соединить отдельные упорядоченные массивы в один. При этом, так как время соединения упорядоченных массивов линейно зависит от их длины, а время сортировки пузырьком - квадратично, то общее время сортировки начального массива будет уменьшено. Причем чем длиннее массив, тем больший выигрыш во времени будет получен.

АРИЗ-Универсал-2010. Задача о полиноме.

- В задаче 7 о вычислении полинома после проведения процедуры частичного свертывания функциональной модели программы была поставлена задача в виде противоречия:
- форма ввода не должна накладывать ограничения, чтобы обеспечить возможность задания произвольного полинома, и должна иметь жесткие ограничения, чтобы проверять корректность введенной строки.
- **Функциональный ИКР**: Модель полинома САМА задается для программы во время ввода пользователем при сохранении произвольности полинома и обеспечении корректности.
- **Противоречия требований:** ЕСЛИ сделать ввод полинома фиксированным, ТО полином будет введен (будет создана модель полинома), НО не будет обеспечена произвольность ввода.
- ЕСЛИ осуществлять ввод полинома в произвольной форме (текстовой строкой), ТО будет обеспечена произвольность ввода полинома, НО проверка ввода и создание модели будут трудны.

АРИЗ-Универсал-2010. Задача о полиноме.

- **Конфликтующие элементы**: Полином и Компоненты формы ввода. Они взаимодействуют между собой, можно менять форму ввода.
- **ОВ** – время ввода полинома
- **ОЗ** – область формы ввода
- **Перечень основных ресурсов системы**: ресурсы полинома, формы ввода, модели полинома.
- **Элепольное решение**: Э1 – полином, Э2 – форма ввода. Между ними необходимо ввести поле взаимодействия П – некоторые правила создания модели полинома.



АРИЗ-Универсал-2010. Задача о полиноме.

- **Ресурсный ИКР**: Икс-элемент из ресурсов системы абсолютно не усложняя систему и не вызывая вредных явлений САМ устраняет необходимость проверки на корректность во время ввода полинома и в пределах области формы ввода, СОХРАНЯЯ возможность введения произвольного полинома.
- **Противоречие свойств:** Конфликтующий элемент должен обладать свойство проверки, чтобы обеспечить корректность ввода, и должен обладать свойством произвольности, чтобы гарантировать ввод любого полинома.
- **ИКР свойств**: Область формы ввода в течение ввода полинома САМА обеспечивает проверку на корректность и гарантирует возможность произвольного полинома.

АРИЗ-Универсал-2010. Задача о полиноме.

- **Возможное решение**. Согласно принципу дробления, структуру полинома следует разделить на части. Естественным образом он разбивается на слагаемые. По принципу заранее подложенной подушки нужно компенсировать ошибки при вводе слагаемых некоторыми средствами формы ввода и модели полинома. На коэффициенты и степени известно ограничение – они могут быть только числами. Это упрощает проверку (раньше при задание всего полинома могли быть введены как числа, так и буквы, обозначающие переменные). Таким образом, следует разбить ввод полинома на ввод слагаемых, проверка каждого из которых будет довольно проста.
- .

Основные результаты работы

- Доказана возможность применения методов ТРИЗ для постановки и решения изобретательских задач в программировании
  - Показано, что АРИЗ 2010 представляет эффективный инструмент решения задач в программировании
  - Проведена свертка системы в программировании с выходом на изобретательскую задачу
- Разработан программный продукт, основанный на алгоритмах постановки изобретательских задач, который автоматизирует вспомогательные процессы и облегчает выполнение этих алгоритмов
  - Проанализированы методы постановки изобретательских задач
  - Адаптирован для программной реализации алгоритм свертки и сопутствующие ему алгоритмы

Наши координаты

Рубин Михаил Семенович, Мастер ТРИЗ:

E-mail: mik-rubin@yandex.ru

Сайт: http://www.temm.ru