Электронный научно-производственный журнал
«АгроЭкоИнфо»

УДК 004.942

Технические аспекты поливариантного расчёта динамических моделей продукционного процесса сельскохозяйственных культур

Медведев С.А.

Агрофизический научно-исследовательский институт

Аннотация

Прикладное использование динамических моделей продукционного процесса требует их выполнения в среде поливариантного расчёта и анализа. Описан ряд технических решений, использованных при разработке подобной среды – системы APEX: основные алгоритмы работы с базой данных, способы хранения данных, используемых конкретными моделями, особенности реализации пользовательского интерфейса, необходимые для достижения необходимой универсальности. Проведён обзор исследований, которые были выполнены с использованием системы APEX, доказывающий высокую эффективность применённых технологических решений.

Ключевые слова: ДИНАМИЧЕСКАЯ МОДЕЛЬ ПРОДУКЦИОННОГО ПРОЦЕССА, ВЫЧИСЛИТЕЛЬНЫЙ КОМПЬЮТЕРНЫЙ ЭКСПЕРИМЕНТ, ADO.NET, КОМПОНЕНТНО-ОРИЕНТИРОВАННОЕ ПРОГРАММИРОВАНИЕ, БАЗЫ ДАННЫХ, ИНТЕГРАЦИЯ

Введение

Модели продукционного процесса в современном сельском хозяйстве позволяют существенно уменьшить трудоёмкость фундаментальных и прикладных исследований [1]. С использованием моделей вместо полевых опытов [2] в ряде случаев можно проводить вычислительные компьютерные эксперименты, получая новую информацию о моделируемой системе за часы, вместо того, чтобы исследовать её в реальности в течение нескольких лет. При этом модели продукционного процесса подразделяются на статистические и динамические [3]. Статистические модели не имеют физического и биологического смысла, они просто интерполируют данные уже проводившихся полевых

Электронный научно-производственный журнал
«АгроЭкоИнфо»

исследований; в связи с этим они не могут давать достоверных прогнозов за рамками тех условий, в которых проводились опыты, результаты которых использовались для составления моделей. В отличие от них, динамические модели основаны на уравнениях реальных процессов, протекающих в моделируемой системе, и поэтому могут быть использованы не только для прогнозов, основанных на интерполяциях, но и в качестве источника новых знаний о моделируемой системе. Поэтому именно динамические модели могут быть использованы в качестве инструмента фундаментальных и прикладных исследований в сельском хозяйстве.

Однако сама по себе динамическая модель продукционного процесса представляет собой вычислительный алгоритм, который способен лишь просчитать динамику развития различных показателей агроэкосистемы для конкретных условий, в то время как для проведения исследований требуется многократно прогнать модель продукционного процесса с различными входными данными и сравнить результаты в разрезе того, что было проварьировано. Это обстоятельство ставит перед модельерами задачу создания некой среды, обеспечивающей вычислительные компьютерные эксперименты с моделями продукционного процесса. Принципиальное решение данной задачи было нами рассмотрено в работе «Методические основы поливариантного расчёта динамических моделей продукционного процесса сельскохозяйственных культур» [4]. Настоящая работа посвящена техническим аспектам решения этой задачи, воплощённого в системе поливариантного расчёта APEX [5].

Постановка задачи

Поскольку задача поливариантного расчёта носит универсальный характер, не зависящий от конкретной модели продукционного процесса, имеется опасность того, что функционал системы станет слишком сложным для конечного пользователя. Чтобы этого не происходило, необходимо подразделить его на две категории: гибкий функционал для настройки системы, который может быть достаточно сложным, но используется редко, и более простой функционал непосредственно для проведения исследований. Иначе говоря, система APEX может рассматриваться двояко:

- как реестр моделей продукционного процесса;
- как среда поливариантного расчёта и анализа.

Функционал реестра предназначен для регистрации в системе поливариантного

Электронный научно-производственный журнал
«АгроЭкоИнфо»

расчёта произвольной модели, удовлетворяющей требованиям совместимости, которые были изложены в работах [4, 5]. Пользователь должен каким-то образом описать структуру данных для модели продукционного процесса, после чего в системе физически создаётся соответствующая структура, а также описать адаптер для модели продукционного процесса, чтобы модель можно было запустить на выполнение. Именно эта функциональность должна поддерживать работу сразу с множеством моделей.

Функционал среды поливариантного расчёта и анализа предназначен собственно для проведения вычислительных компьютерных экспериментов с теми моделями, которые были зарегистрированы в базе данных: создание сценариев и проектов, связывание сценариев с результатами модельных расчётов и анализ результатов в разрезе проварьированных данных.

Для того, чтобы совместить гибкость системы с интуитивной понятностью, при разработке следует использовать подход, основанный на типовых решениях. При этом подходе большая часть функциональности стандартизуется таким образом, чтобы вписаться в небольшое количество типовых форм. При этом техническая реализация такова, что типовые формы реализуются с помощью стандартных механизмов, реализованных в ядре системы. Это позволяет не только стандартизировать интерфейс, делая его более однотипным и легко осваиваемым для пользователя, но и стабилизировать качество тех функциональных элементов, которые на нём основаны, т.к. типовое решение разрабатывается и отлаживается один раз, а потом многократно используется.

Материалы и методы

Требования к системе поливариантного расчёта подразумевают хранение в единой реляционной структуре данных, относящихся к информационным доменам разного уровня абстракции. Так, «модель», «проект», «сценарий» и пр. – это понятия, которые известны на момент разработки системы и не могут меняться действиями конечного пользователя. В связи с этим для хранения этих данных могут применяться обыкновенные реляционные таблицы. Но внутри любой подключаемой модели может быть любая своя структура данных, и всё, что мы можем о ней сказать, – это то, что её можно разбить по предопределённым факторам. Эта часть модели данных может меняться пользователем. Для решения такой задачи нами было предложено оригинальное решение, основанное на использовании системных представлений INFORMATION_SCHEMA, подробно

описанное в работе [6]. Оно состоит в том, что как таблицы, структура которых известна на этапе разработки системы поливариантного анализа, так и таблицы, структура которых зависит от конкретной модели и задаётся пользователем при настройке системы под конкретную модель, являются физическими таблицами базы данных, а связь между ними осуществляется посредством специальных представлений и функций, берущих одну часть данных из системных представлений INFORMATION_SCHEMA, а другую – из физических таблиц. Это позволяет, с одной стороны, не терять в производительности, как это происходит при использовании традиционного для таких задач подхода с «метатаблицами» [7], а с другой, – поддерживать целостность данных и метаданных. Кроме того, этот подход позволяет оптимизировать сам процесс разработки, используя ORM (Object-Relation Mapper) для доступа не только к постоянным таблицам системы (проект, модель, сценарий), но и к представлениям, отображающим метаданные системы (корневые и дочерние таблицы модели, структура таблиц результатов моделей и пр.).

Данные, используемые конкретными моделями продукционного процесса, хранятся в системе следующим образом. Каждому предопределённому фактору в зарегистрированной модели соответствует одна корневая таблица. Запись в корневой таблице соответствует одной градации фактора одной модели. Такие таблицы содержат по две колонки: автоматически генерируемый идентификатор и текстовое описание градации фактора. Все свои данные модели продукционного процесса хранят в дочерних таблицах, которые привязаны к определённому фактору через внешний ключ на корневую таблицу. Если дочерняя таблица содержит именно табличные данные, то она должна иметь идентификатор, уникальный в пределах одного набора записей, используемого моделью при прогоне. Поскольку в базе данных системы APEX все эти записи связаны с записью корневой таблицы, первичный ключ в таких таблицах должен быть составным и включать в себя как внешний ключ на корневую таблицу, так и определённый пользователем идентификатор в пределах одной градации фактора. Если такой идентификатор не выбран, то первичный ключ в дочерней таблице одновременно является внешним ключом на корневую, что обеспечивает связь «один к одному», и для каждого прогона модели можно получить только скалярные значения. Кроме входных данных, каждая зарегистрированная модель содержит таблицу результатов. Поскольку система поливариантного расчёта предназначена для прогона множества сценариев расчёта, эта таблица должна быть связана со сценариями связью «один ко многим». Кроме этой связи,

она должна содержать колонку для хранения идентификатора времени, уникальную в пределах каждого сценария, и поля вектора состояния модели (рис. 1).

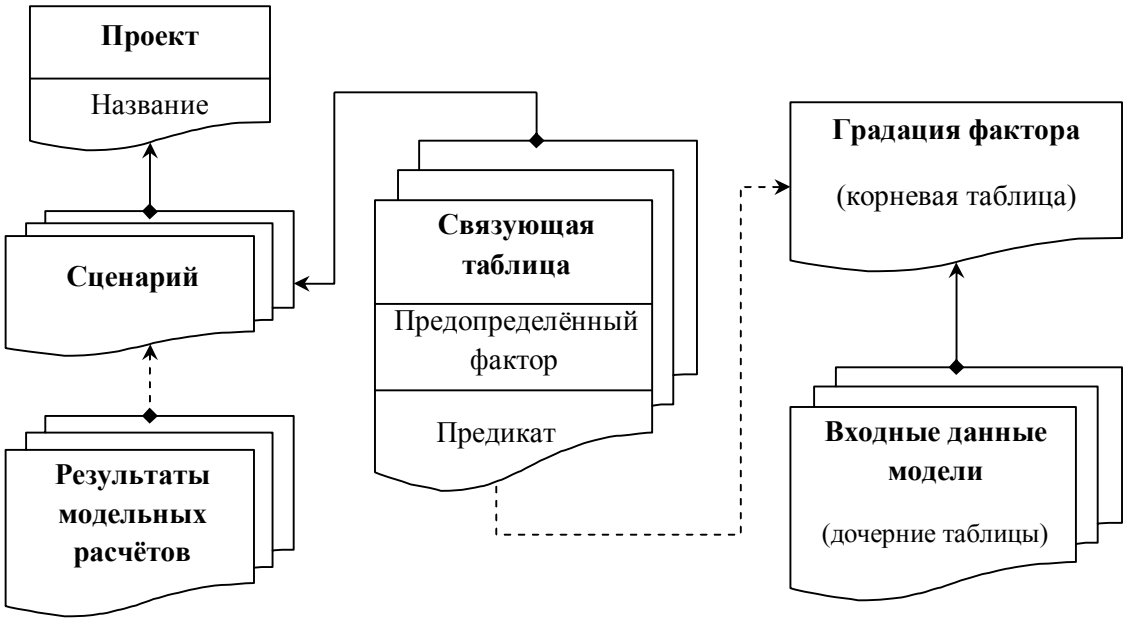

Рис. 1. Принципиальная схема базы данных системы поливариантного расчёта APEX

Во избежание конфликтов имён при создании пользовательских таблиц в системе поливариантного расчёта используются префиксы, отделяемые от основного имени таблицы знаком подчёркивания. Каждой модели соответствует свой префикс, длина которого, по соглашению, не превышает трёх символов. Для постоянных таблиц, а также представлений, функций и ограничений используется префикс «reg», который запрещено использовать в таблицах, создаваемых пользователем. Такие же префиксы используются для предотвращения конфликтов имён таблиц, относящихся к разным предопределённым факторам внутри одной модели, а также конфликтов имён колонок, явно создаваемых пользователем, и колонок, автоматически добавляемых системой (например, внешнего ключа на корневую таблицу во всех дочерних или ссылки на сценарий в таблице результатов). Всё это позволяет гарантировать, что никаких конфликтов имён между системными и пользовательскими объектами возникнуть не может. Для того, чтобы пользователь при работе с системой не видел эти префиксы, там предусмотрены специальные постоянные таблицы с описаниями таблиц и колонок, создаваемых пользователем; эти описания отображаются на элементах пользовательского интерфейса вместо системных названий с префиксами.

Электронный научно-производственный журнал
«АгроЭкоИнфо»

Кроме самой базы данных, для решения стоящих перед системой поливариантного расчёта задач необходимо клиентское приложение, предоставляющее пользовательский интерфейс к базе данных. Оно тоже содержит ряд оригинальных технических решений. Для начала сформулируем общие требования к техническим решениям и сопоставим с ними требования к используемой для реализации этих решений технологии (табл. 1).

Таблица 1. Требования к технологиям, которые должны использоваться для разработки системы поливариантного расчёта

Требования к приложению	Требования к технологии
1. Приложение должно предоставлять возможность одновременно работать как с постоянными таблицами, так и с таблицами, созданными пользователем	Технология должна предоставлять общий механизм подключения к базе данных, на основе которого работает как ORM, так и структуры данных, задаваемые с помощью логики
2. Приложение должно иметь расширяемую архитектуру для подключения исполняемых модулей моделей продукционного процесса	Технология должна предоставлять удобный инструмент для подключения внешних программных модулей к развёрнутому приложению
3. Приложение должно предоставлять универсальный пользовательский интерфейс для доступа к разным таблицам	Технология должна позволять автоматически подстраивать пользовательский интерфейс под объектную модель

Этим требованиям хорошо отвечает платформа Microsoft.NET. Она является широко используемой платформой для построения коммерческих приложений [8]. Для подключения к базе данных в рамках этой платформы реализована технология ADO.NET, которая состоит из двух частей: компоненты сущностей и управляемые поставщики [9]. Управляемые поставщики – это набор интерфейсов, описывающих подключение к базе данных, sql-команды, транзакции и курсоры для чтения данных. Эти интерфейсы реализованы для всех популярных СУБД, позволяя отделить бизнес-логику приложений от конкретной СУБД. Вторая часть технологии ADO.NET – компоненты сущностей – представляет собой объектную модель произвольных таблиц с отношениями между ними. С помощью этой модели можно работать как со структурой данных, так и с самими данными, что является оптимальным решением при работе с таблицами, определёнными пользователем.

Для подключения внешних программных модулей платформа Microsoft.NET предоставляет ещё более мощные средства. Все сборки .NET содержат не только

исполняемый код, но и описание его структуры, позволяющее получать полную информацию обо всех типах на этапе выполнения программы с помощью механизма ***рефлексии*** [10]. Кроме того, платформа содержит в себе динамический загрузчик сборок, позволяющий загружать сборки .NET на этапе выполнения программы. Всё это позволяет реализовать плагинную архитектуру достаточно легко. Система поливариантного расчёта должна содержать сборку, в которой объявлен интерфейс, который должны реализовывать классы в подключаемых модулях. Во время выполнения она должна искать сборки плагинов, динамически их загружать, искать в них такие классы посредством рефлексии и запускать на выполнение. Это позволяет расширять функциональность приложения, которое было один раз написано и развёрнуто без перекомпиляции.

Для подстройки пользовательского интерфейса под объектную модель платформа Microsoft.NET предоставляет так называемую компонентную модель [11]. Все инструменты построения пользовательского интерфейса, входящие в эту платформу: ASP.NET для Web-сайтов, Windows Forms для стандартных оконных приложений и WPF для приложений со сложной графикой, – базируются на единой компонентной модели. Она предоставляет базовые классы, описывающие редактируемые объекты, которые позволяют по этому описанию автоматически формировать элементы пользовательского интерфейса. По умолчанию реализация этих классов основана на рефлексии, но может быть переопределена. В частности, компоненты сущностей ADO.NET имеют свою реализацию компонентной модели, что позволяет использовать динамически сформированные структуры данных так же, как описание типов, получаемое через рефлексию. В системе поливариантного расчёта это позволяет работать как с постоянными таблицами системы, так и с пользовательскими, причём для работы с пользовательскими таблицами достаточно одной формы, которая динамически подстраивается под структуру редактируемых таблиц.

В рамках архитектуры, использующей компонентную модель Microsoft.NET, одним из оригинальных технических решений является редактор адаптеров подключенных моделей. Поскольку адаптеры по семантике являются плагинами, система не может делать никаких предположений относительно используемых в этих адаптерах типах и структурах данных для настройки. В связи с этим для настройки адаптера требуется элемент пользовательского интерфейса, который бы подстраивался под объектную модель автоматически. В то время как для отображения табличных данных платформа

Microsoft.NET предоставляет готовое решение, для работы с адаптерами потребовался инструмент для представления данных иной структуры. Он выполнен с помощью стандартного элемента управления Windows Forms «дерево», предоставляющего достаточно удобный API для работы с узлами. Решение состояло в том, чтобы средствами компонентной модели Microsoft.NET проанализировать структуру объектной модели, которая редактируется через дерево, и динамически сгенерировать дерево по определённым правилам. При этом для управления отображением разных объектов по возможности также используются интерфейсные решения компонентной модели. Данный компонент используется совместно с панелью свойств – стандартным элементом управления Windows Forms, автоматически генерирующим редакторы для всех свойств переданного ему объекта. Этому элементу управления передаётся объект, связанный с выделенным пользователем узлом дерева. Поскольку структура дерева, с которым работает пользователь, полностью определяется объектной моделью, для внесения изменений в функционирование редактора требуется внести их только в редактируемую объектную модель. Никаких изменений на уровне дизайнера форм не требуется. Кроме того, это решение может быть использовано повторно. В частности, большинство форм системы, в которых используется элемент управления «дерево», использует один и тот же один раз разработанный компонент, который связывает этот элемент управления с произвольной иерархической объектной моделью. Кроме редактора адаптеров, это решение используется в редакторе структуры данных регистрируемой модели.

Для сохранения объектной модели структуры данных для модели в файл используется встроенный сериализатор платформы Microsoft.NET, позволяющий автоматически сохранять и читать объекты разных типов в потоки ввода-вывода; разработчику приложения достаточно указать, какие свойства или поля классов подлежат сериализации, требуемая же структура файла создастся автоматически. Для хранения структур данных моделей продукционного процесса используется расширение bmod, которое при установке ассоциируется с программой APEX. При этом, если программа уже запущена, файл, открытый из проводника Windows, будет открыт в уже запущенной копии программы. Для реализации такого поведения новая копия программы ищет уже запущенную и, если находит, передаёт ей с помощью специального сообщения Windows путь к файлу, запустившему её, после чего завершается, не создавая главного окна. Главное окно уже запущенной копии программы обрабатывает это сообщение и открывает файл.

Электронный научно-производственный журнал
«АгроЭкоИнфо»

После того, как структура данных для модели продукционного процесса сформирована посредством элемента управления «дерево», её нужно зарегистрировать в базе данных. Этот процесс включает в себя две основные операции.

1.Физическое создание всех пользовательских таблиц в базе данных: корневых, дочерних и таблицы результатов.

2.Заполнение данными таблиц, в которых хранится расширение метаинформации: список корневых таблиц модели и описания колонок.

В то время как вторая операция является тривиальной манипуляцией с данными посредством ORM, первая требует нетривиального решения. Дело в том, что структура входных и выходных данных в конкретной модели продукционного процесса не является чем-то один раз и навсегда определённым. Модель, как и любой программный продукт, развивается, и требования к входным и выходным данным могут меняться. Однако данные, с которыми пользователь уже работал, как правило, не должны теряться после повторной регистрации модели. В связи с этим для внесения изменений в структуру данных должна быть реализована возможность «мягкого» обновления базы данных, т.е. не приводящего к потере данных в существующих таблицах.

Как правило, эта операция поддерживается самими ORM. Однако там она рассчитана на то, что структура данных в системе не меняется пользователем, и, таким образом, инструмент мягкого обновления является исключительно инструментом разработчика; а в системе поливариантного расчёта структура БД меняется пользователем. Более того, работа с таблицами, создаваемыми пользователем, посредством ORM невозможна, т.к. эта технология предполагает, что все сущности известны на момент разработки системы. В связи с этим в самой системе поливариантного расчёта должен быть реализован собственный алгоритм «мягкого» обновления. Кроме того, учитывая, что модели в БД системы разделяются по префиксам, необходим нестандартный алгоритм, позволяющий обновить только часть БД, соответствующую одной модели.

Этот алгоритм таков. Пользователь при редактировании структуры данных для модели задаёт такие имена таблиц и колонок, какие требуются самой модели. При регистрации описание этой структуры сначала конвертируется в экземпляр класса DataSet – центральный класс компонентов сущностей ADO.NET, представляющий набор таблиц, связанных между собой. При этом к формируемым таблицам добавляются все системные

поля и префиксы. После этого с помощью запросов к INFORMATION_SCHEMA из БД получается информация о таблицах, имеющих такой же префикс, как и у регистрируемой модели, а также об их отношениях. После этого запускается операция сравнения двух экземпляров DataSet, которая проверяет, какие появились или удалились таблицы, колонки, первичные и внешние ключи. Эта операция заполняет список DDL-команд. Каждый раз, когда обнаружилось расхождение, в список добавляется команда, содержащая скрипт этой команды и тип расхождения. После этого команды сортируются по типу в следующем порядке:

1. Удаление внешнего ключа
2. Удаление уникального ключа
3. Удаление первичного ключа
4. Удаление таблицы
5. Удаление колонки
6. Создание колонки
7. Изменение колонки
8. Создание первичного ключа
9. Создание уникального ключа
10. Создание внешнего ключа

После этого все отсортированные команды выполняются друг за другом. Для того, чтобы отделить алгоритмическую часть от конкретной СУБД, имеется два интерфейса, расширяющих список интерфейсов управляемых поставщиков. Первый интерфейс предназначен для получения из БД результатов запросов к INFORMATION_SCHEMA, из которых потом формируется экземпляр DataSet, представляющий структуру данных. На уровне реализации этого интерфейса могут быть учтены особенности представлений INFORMATION_SCHEMA в конкретных СУБД и каких-нибудь отклонений синтаксиса от стандарта. Этот интерфейс поддерживает фильтрацию по префиксу, требуемую для реализации специфических требований к мягкому обновлению БД. Второй интерфейс предназначен для того, чтобы для каждого типа расхождения в структуре данных сгенерировать нужную DDL команду. Он также позволяет учесть особенности синтаксиса SQL в конкретной СУБД, а, кроме того, некоторые неочевидные особенности её реализации (например, в SQL Server нельзя одним запросом изменить свойство Identity у колонки, для этого колонку требуется пересоздавать).

Электронный научно-производственный журнал
«АгроЭкоИнфо»

Работа с таблицами, которые при этом создаются или обновляются, также отвязана от конкретной СУБД через интерфейсы управляемых поставщиков ADO.NET. Программа всегда работает с подключениями, командами, транзакциями и курсорами посредством этих интерфейсов, а не их реализаций в конкретных провайдерах, и лишь при инициализации задаётся, что работа будет производиться с СУБД Microsoft SQL Server. Единственным местом, которое не покрывают интерфейсы провайдеров, является различие в синтаксисе SQL в разных СУБД. Для того, чтобы обойти это, система поливариантного расчёта содержит синтаксическую объектную модель запроса и интерфейс, через который запросы, представленные в этой объектной модели, конвертируются в текст запросов к конкретной СУБД.

После регистрации модели в базе данных пользователь должен настроить её адаптер. Технически подключение адаптера реализуется через интерфейс IModelAdapter, описанный в сборке Registrator.Exchange. Он содержит три метода и два свойства. Для того, чтобы адаптер работал, необходимо реализовать метод Run, а также оба свойства. Свойство CanRun должно возвращать true, если адаптер настроен, и модель может быть запущена, и false в противном случае; свойство CanFill, если методы Fill и GetFillDescrition не реализованы, должно возвращать false. Метод Run в качестве параметров принимает набор входных данных для единичного прогона модели в виде словаря, ключом которого является тип предопределённого фактора (перечисление), а значением – ADO.NET DataSet, идентификатор этого прогона и индикатор выполнения, а возвращает набор результатов модели в виде ADO.NET DataTable. Методы Fill и GetFillDescription можно реализовать, если модель берёт данные для единичного расчёта из какого-нибудь хранилища, чтобы можно было их импортировать оттуда в базу данных системы поливариантного расчёта. Очевидно, что такой интерфейс адаптера очень прост и в случае хорошей технической и семантической совместимости требует минимальных действий от разработчика, которому необходимо интегрировать модель продукционного процесса в систему поливариантного расчёта. Более того, он позволяет оборачивать модель именно «снаружи», не меняя её внутренней организации.

Для подключения адаптера модели сборка, в которой объявлен нужный адаптер, должна находиться в директории программы. При старте программа автоматически сканирует директорию установки, загружает все доступные сборки и ищет в них типы, реализующие интерфейс адаптера. Эти типы должны быть сериализуемыми с

использованием BinaryFormatter, т.к. настройки каждого адаптера хранятся в файловой системе в виде бинарного файла. После этого найденные типы появляются в меню, из которого можно выбрать адаптер для любой модели, зарегистрированной в БД, и настроить его с использованием древовидного редактора с панелью свойств. Это решение позволяет подключать к единожды установленной программе произвольные модели продукционного процесса, удовлетворяющие требованиям совместимости [12]. Кроме того, поддержка составных адаптеров позволяет не писать их целиком, а только подменять отдельные блоки.

Результаты

Система поливариантного расчёта APEX с подключенной к ней моделью продукционного процесса третьего уровня продуктивности AGROTOOL с успехом была применена в ряде исследований как фундаментально-теоретического, так и прикладного характера. В первую очередь, это задачи идентификации параметров, оптимизации агротехнологий, прогноза урожайности в условиях климата будущего, а также оперативное сопровождение и проактивное управление полевым опытом или производственным посевом. Решение всех этих задач подробно описано в соответствующих публикациях [13-21].

Идентификация параметров – это процедура настройки модели продукционного процесса, нацеленная на повышение достоверности получаемых результатов модельных расчётов. Для этой процедуры необходимы данные реальных полевых опытов, с которыми сверяются результаты поливариантных модельных расчётов, в которых проварьированы эвристические параметры модели, не имеющие физического смысла [13, 14]. В частности, с использованием системы поливариантного расчёта APEX были идентифицированы блоки азотного питания и развития. Блок азотного питания основан на эвристическом уравнении, которое содержит в себе четыре коэффициента, подлежащих идентификации. Они были проварьированы, а невязка (отличие от результатов полевых опытов) определялась по урожаю.

Блок развития в модели AGROTOOL основан на понятии фазы развития. Так, для злаков принято выделять следующие фазы: всходы, третий лист, кущение, выход в трубку, колошение, цветение, молочная спелость, восковая спелость, полная спелость. Кроме того, развитие растения описывается специальным показателем «физиологическое

время», который представляет собой количественное выражение степени зрелости растения. В модели AGROTOOL значение 0 соответствует началу вегетации, значение 1 – цветению, значение 2 – полной спелости. Для каждой фазы развития имеется пороговое значение физиологического времени $T_{Ph}(IPh)$, по достижении которого эта фаза наступает. Это пороговое значение является параметром, подлежащим идентификации. Поскольку этот параметр – свой для каждой фазы, для его полной идентификации требуется по очереди проварьировать его значения для всех фаз исследуемой культуры. В результате вычислительных компьютерных экспериментов были подобраны значения этих коэффициентов, при которых невязка была минимальной [15].

Задача оптимизации агротехнологий является традиционной для масштабных полевых опытов. Использование вычислительного компьютерного эксперимента с моделью продукционного процесса позволяет подобрать оптимальные параметры агротехнологий с гораздо меньшей трудоёмкостью. В случае использования регрессионных моделей возможно аналитическое решение задачи оптимизации урожая, но в случае динамической модели, представляющей собой сложный алгоритм, возможен лишь поиск квази-оптимального решения прямым перебором проварьированных параметров агротехнологий. С помощью системы APEX и модели продукционного процесса AGROTOOL проводилась оптимизация полива пшеницы сорта Саратовская-29 в засушливых регионах Поволжья. Рассматривался режим полива вида «если почва высохла ниже критического значения u_{min}, полить до достижения влажности u_{max}»; при этом оптимизация производилась не по величине урожая, а по экономическому критерию: из прибавки урожая, умноженной на цену зерна, вычитались затраты на полив. Было получено, что оптимальная стратегия орошения должна характеризоваться достаточно низким (близким к влажности завядания) значением величины u_{min}. При этом «триггер» включения полива будет срабатывать только по достижении критического уровня влагообеспеченности, могущего привести к значительному снижению конечной продуктивности, а в условиях умеренного стресса агроэкосистема будет ждать «естественного» орошения осадками, экономя таким образом затраты на воду и накладные расходы [16].

Исследование агросистемы в условиях климата будущего – это типичная задача динамического моделирования. Решать эту задачу с использованием регрессионных моделей заведомо некорректно, потому что при исследовании моделируемого объекта в

условиях климата будущего мы находимся за пределами условий, в которых была получена модель. Для проведения этой работы в базу данных системы APEX были загружены сгенерированные методом Монте-Карло суточные погодные метеоданные, основанные на интегральных характеристиках климата будущего, полученных с помощью моделей ECHAM и Hadley, после чего эти данные использовались для модельных расчётов моделью AGROTOOL. Серия вычислительных компьютерных экспериментов показала, что при сохранении современных сроков сева урожайность культур в условиях климата будущего будет падать, но при более ранних сроках сева урожайность будет возрастать, и скорость созревания также увеличится [17].

Также с помощью системы APEX была создана карта урожайности в ландшафтном стационаре Губино. Описания градаций фактора «Почва» содержали информацию о географической привязке соответствующих почвенных контуров, а данные брались из почвенного реестра ландшафтного стационара. После выполнения модельных расчётов для большого количества лет результаты статистической обработки по урожаю в разрезе проварьированного предопределённого фактора «Почва» экспортировались, и с помощью географической привязки эти данные использовались для составления легенды карты [18].

Оперативное сопровождение – это технология проведения модельных расчётов одновременно с полевым опытом или производственным посевом. При этом, в отличие от исследований, которые заранее требуют полного набора суточных погодных метеоданных за весь вегетационный период, данные в системе APEX постоянно актуализировались свежими данными метеорологических наблюдений, а «веер» погоды для будущего отрезка вегетационного периода генерировался с использованием метода Монте-Карло. Это позволяло осуществлять динамически уточняющийся прогноз урожайности и сроков наступления фенофаз для тут же проводящегося полевого опыта. Наличие «веера» позволяет оценить прогнозируемый урожай в повторностях, с использованием реальной дисперсии погоды за предшествующий период наблюдений [19, 20].

Эта технология может быть использована для определения оптимальной даты проведения какого-нибудь агротехнического мероприятия, в частности, азотной подкормки по листу. С использованием динамической модели продукционного процесса AGROTOOL и системы поливариантного расчёта APEX проводились исследования эффективности трёх различных подходов к управлению режимов азотных подкормок: декларативный (определяется конкретная дата проведения подкормки, оптимальная для

всех лет в целом), реактивный (подкормка производится при достижении агросистемой определённого состояния) и проактивный (с помощью технологии оперативного сопровождения сравнивается урожай для случаев, когда подкормка производится сегодня, и когда подкормка производится завтра; подкормку надо сделать в тот день, когда «сегодня» становится лучше, чем «завтра»). В результате было получено, что декларативный подход в отдельные годы может быть крайне неэффективным; эффективность реактивного подхода зависит от использованного предиката, и наилучшего результата удалось добиться, если в качестве предиката взять физиологическое время; проактивное управление показало ещё более высокую эффективность. Хотя даже при реактивном управлении по физиологическому времени эффективность управления приближалась к максимальной, как если бы мы заранее знали оптимальную дату, точное определение физиологического времени в полевых условиях затруднительно. Этого недостатка лишено проактивное управление, в связи с чем именно эта методика поддержки принятия решений в сельском хозяйстве является наиболее перспективной в плане практической эффективности [21].

Выводы

Проведённые исследования наглядно демонстрируют работоспособность технологических решений, использованных при разработке системы поливариантного расчёта APEX. Механизм поливариантного расчёта может быть применён для большого количества фундаментально-теоретических и прикладных задач. Именно в рамках универсальной среды поливариантного расчёта модель может быть эффективно использована в практических целях. Кроме того, из тех сложностей, с которыми мы столкнулись при проведении исследований, вытекает ряд перспективных направлений развития системы поливариантного расчёта. Например, для ускорения пакетных расчётов имеет смысл в распараллеливании вычислений. В настоящий момент система APEX не поддерживает распараллеливание, кроме того, сама модель AGROTOOL, которая использовалась при проведении описанных исследований, разработана без учёта возможности распараллеливания, так что запуск нескольких её экземпляров одновременно приводит к сбоям и ошибкам. Другое перспективное направление работы – это использование чужих моделей. По этой технологии уже подключены сторонние модели, в частности, модель MONICA, разработанная в Zalf [22], также проведено исследование возможности

подключения семейства моделей, написанных для универсальной среды GUICS [23]. Однако тот факт, что при проведении конкретных исследований система поливариантного расчёта APEX периодически дорабатывалась без переделывания базовой функциональности, свидетельствует о том, что её архитектура является расширяемой, и дальнейшая работа в этом направлении не столкнётся с принципиальными неразрешимыми проблемами.

Список использованных источников

1. Полуэктов Р.А. Динамические модели агроэкосистемы. – Л.: Гидрометеоиздат. – 1991. – 312 с.

2. Доспехов Б. А. Д 70 Методика полевого опыта (с основами статистической обработки результатов исследований). – 5-е изд., доп. и Перераб. - М.: Агропромиздат, 1985. – 351 с.

3. Полуэктов Р.А., Смоляр Э.И., Терлеев В.В., Топаж А.Г. Модели продукционного процесса сельскохозяйственных культур // Изд-во СПбГУ. – 2006. – 390 с.

4. Медведев С.А. Методические основы поливариантного расчёта динамических моделей продукционного процесса сельскохозяйственных культур // АгроЭкоИнфо. – 2015, №4. http://agroecoinfo.narod.ru/journal/STATYI/2015/4/st_16.doc.

5. Медведев С.А. Система автоматизации компьютерного многофакторного эксперимента с динамическими моделями продукционного процесса // Материалы международной школы-семинара «Фундаментальные и прикладные исследования в математической экологии и агроэкологии», Барнаул, 22-24 июня, 2012. – Барнаул: Изд-во Алт. ун-та. – 2012. – С. 98-104.

6. Медведев С.А. Перспективы создания универсальной оболочки поливариантного расчета моделей продукционного процесса // Материалы Всеросс. конф. (с международ. участ.) «Математические модели и информационные технологии в сельскохозяйственной биологии: итоги и перспективы», 14-15 октября 2010 г., Санкт-Петербург. – СПб.: АФИ. – 2010. – С. 264-269.

7. Левин А. Метаданные как основа баз данных мониторинга. Экоцентр МТЭА. – 2002.

8. Дубровцев А. Microsoft .NET в подлиннике. BHV-СПб. – 2004. – 704 с.

9. Сеппа Д. Программирование на Microsoft ADO.NET 2.0. Мастер-класс. – Питер. – 2007. – 784 с.

10. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft . NET Framework 2.0 на языке C#. – Питер. – 2006. – 656 с.

11. Агуров П.С. Разработка компонентов в MS Visual Studio 2005/2008. BHV-СПб. – 2008. – 480 с.

12. Гамма Э., Хелм Р., Джонсон Р., Влиссидес Дж. Приемы объектно-ориентированного проектирования. – Питер,. – 2005. – 366 с.

13. Хворова Л.А., Топаж А.Г. Построение моделей агроэкосистем и их адаптация к конкретным условиям // Научно-технические ведомости СПбГТУ. – 2011, №115. – С. 99-105.

14. Poluektov R.A., Oparina I.V., Zakharova E.T., Fintushal S.M. Parameter Estimation of Soil-Plant-Atmosphere Model // Agrophysical and Ecological Problems of Agriculture in the 21-th Century. – SPb. – 2004, vol. 4, p. 11-17.

15. 1. Льюнг Л. Идентификация систем. Теория для пользователя. – М.: «Наука». – 1991. – 432 с.

16. Медведев С.А., Полуэктов Р.А., Топаж А.Г. Оптимизация стратегии орошения с использованием методов поливариантного анализа динамики агроэкосистем // Мелиорация и водное хозяйство. – 2012, № 2. – С.10-13 (из Перечня ВАК РФ).

17. Полуэктов Р.А., Топаж А.Г., Якушев В.П., Медведев С.А. Использование динамической модели агроэкосистемы для оценки влияния климатических изменений на продуктивность посевов (теория и реализация) // Вестник Российской академии сельскохозяйственных наук. – 2012, №2. – С. 7-12.

18. Интеграция системы поливариантного расчета динамических моделей продуктивности АРЕХ с ГИС на примере ландшафтного стационара «Губино». Баденко В. Л., Медведев С. А. // Материалы научной сессии по итогам 2013 года Агрофизического института. – СПб.: АФИ. – 2014. – С. 49-52.

19. Медведев С.А., Топаж А.Г., Белов А.В., Глядченкова Н.А., Лекомцев П.В. Распределенный измерительно-моделирующий комплекс для оперативного сопровождения полевого опыта // АгроЭкоИнфо. – 2015, №2. http://agroecoinfo.narod.ru/journal/STATYI/2015/2/st_08.doc.

20. Использование измерительно-моделирующего комплекса для оперативного модельного сопровождения полевых опытов на Меньковской опытной станции. Меедведев и др. Материалы научной сессии по итогам 2013 года Агрофизического института. – СПб.: АФИ. – 2014. – С. 53-56.

21. Топаж А.Г., Медведев С.А., Захарова Е.Т. Проактивное управление в компьютерных системах поддержки агротехнологических решений на примере управления азотными подкормками // АгроЭкоИнфо. – 2016, №4. http://agroecoinfo.narod.ru/journal/STATYI/2016/4/st_425.doc.

22. Claas Nendel. MONICA: A Simulation Model for Nitrogen and Carbon Dynamics in Agro-Ecosystems. In: Mueller L., Saparov A., Lischeid G. (eds) Novel Measurement and Assessment Tools for Monitoring and Management of Land and Water Resources in Agricultural Landscapes of Central Asia. Environmental Science and Engineering. Springer, Cham.

23. Acock B., Pachepsky Ya. A., Mironenko E.V., Whisler F. D., Reddy V.R. GUICS: A Generic User Interface for On-Farm Crop Simulations // Agronomy J. – 1999. – No 91. – P. 657–665.

Электронный научно-производственный журнал
«АгроЭкоИнфо»

Цитирование:

Медведев С.А. Технические аспекты поливариантного расчёта динамических моделей продукционного процесса сельскохозяйственных культур // АгроЭкоИнфо. – 2018, №1. – http://agroecoinfo.narod.ru/journal/STATYI/2018/1/st_113.doc.