УДК 519.95: 518.0: 621.391: 681.325

В. П. Зинченко, В. В. Борисов, Д. И. Конотоп

АНАЛИЗ СРЕДСТВ И МЕТОДОВ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ СИНТЕЗА СТРУКТУР КОНЕЧНО–ЭЛЕМЕНТНЫХ МОДЕЛЕЙ

Введение

Основной тенденцией развития современной промышленности является реорганизация и реструктуризация опытно−конструкторских работ и производства на основе новых концепций, идей и информационных технологий (ИТ). При этом одной из главных задач является повышение качества и сокращения сроков проектирования технических объектов (ТО). Одним из наиболее важных, трудоемких и сложных видов проектных задач является определение напряженно−деформированного состояния (НДС) несущих конструкций ТО. Он необходим для разработки структуры и определения оптимальных параметров элементов конструкции проектируемого ТО. При этом наиболее эффективным методом исследования НДС является метод конечного элемента (МКЭ), который позволяет учитывать влияние упругих деформаций конструкций при расчете НДС элементов силового набора. МКЭ позволяет решать задачи определения НДС для конструкций любой конфигурации и сложности. При этом МКЭ хорошо поддается автоматизации и поэтому является эффективным инструментом решения задачи оптимизации механических свойств конструкций сложных ТО (СТО).

При формировании конечно−элементной модели (КЭМ) конструкции СТО возникает проблема снижения времени моделирования, так как наилучшие результаты получаются при использовании КЭМ, моделирующих конструкцию СТО в целом. Наиболее эффективно эта проблема решается с помощью автоматизированных ИТ, для реализации которых требуется описание алгоритмов решения задач, на основании которых разработываются методы и средства.

В статье проведен анализ функциональных возможностей различных систем управления базами данных (СУБД) и, в частности, объектных СУБД (ОСУБД). Проанализированы их преимущества и недостатки. Показано, что наиболее оптимально задача автоматизированного формирования КЭМ решается с помощью объектных технологий управления данными.

Рис. 1. Схема взаимодействия алгоритмов и данных в ПМ

Анализ проблемы

В современных расчетных программных комплексах, таких как MSC. Nastran, Ansys и др., которые используют МКЭ в качестве основного метода расчета НДС, проблема автоматизированного формирования КЭМ конструкции СТО решается путем использования геометрических моделей (ГМ), состоящих из отдельных фрагментов, чья конфигурация определяет количество и расположение узлов КЭ. Каждому фрагменту ГМ соответствует определенная конструктивная зона или элемент конструкции. Такая технология позволяет использовать для всех конструктивных зон единый универсальный алгоритм формирования КЭМ. Типы и параметры КЭ задаются в интерактивном режиме, отдельно для каждой моделируемой конструктивной зоны. Вне зависимости от того, где формируется ГМ, ее формирование осуществляется в интерактивном режиме, с использованием системы меню [1, 2]. Интерактивный режим не дает возможности моделировать все элементы конструкции, воспринимающие и передающие нагрузки, что снижает качество расчета. На основании опыта проектирования установлено, что время, необходимое для формирования ГМ в интерактивном режиме, примерно пропорционально квадрату количества $N_Э$ фрагментов, входящих в ее

структуру: $t_{ГМ} \approx \lambda N_{Э}^{2}$, где λ – коэффициент пропорциональности. В результате время, необходимое для формирования КЭМ конструкции СТО, например, планера транспортного самолета, увеличивается до 2–4 лет, что неприемлемо с экономической точки зрения.

После формирования ГМ производится последовательное формирование локальных КЭМ для всех ее фрагментов. Затем локальные КЭМ объединяются в общую модель. Объединение производится как в автоматическом, так и в интерактивном режиме. В автоматическом режиме используется универсальный метод, основанный на объединении узлов, расстояние между которыми меньше заданного. Такой метод не может считаться надежным, поскольку структуры локальных КЭМ формируются независимо. Поэтому существует вероятность того, что часть узлов не будут объединены, в частности вследствие неточностей, допущенных при формировании ГМ, или вследствие сложной конфигурации ГМ. В результате снижается качество анализа НДС.

Анализ нерешенных вопросов

Для объединения нескольких КЭМ в единую модель требуется, чтобы в зоне стыка присутствовали узлы с одинаковыми координатами. Взаимодействие КЭМ возможно только тогда, когда в их структурах имеются КЭ с одинаковыми номерами узлов. Для объединения локальных КЭМ в общую модель (синтеза структуры КЭМ) требуется информация о локальных номерах узлов, подлежащих объединению. Такая информация может быть получена только с помощью алгоритмов, учитывающих особенности функционирования моделируемых конструкций, что возможно только при высокой специализации ПО. В результате количество вариантов алгоритмов, необходимых для решения автоматизированного формирования КЭМ, становится сопоставимым с количеством объектов в БД.

Характерной особенностью современных ИТ, использующих файловые системы, является логическое разделение данных и алгоритмов: данные хранятся в файлах, а алгоритмы в программных модулях (ПМ). Связь между алгоритмами и данными в таких системах устанавливается непосредственно в процессе решения задачи (рис. 1), что приводит к увеличению вероятности возникновения ошибок решения проектных задач [3, 4]. Кроме того возникает проблема производительности, т.к. ПМ не входят в структуру баз данных (БД), а значит находятся вне зоны обслуживания СУБД.

Выбор алгоритмов является частью процесса проектирования, поскольку проектные задачи относятся к классу обратных задач. По этой причине перечень используемых алгоритмов не может быть определен заранее. В результате возникает необходимость выбора и подключения

алгоритмов непосредственно в процессе решения задачи (динамического подключения алгоритмов), которая не может быть решена средствами современных PDM–систем, поскольку в них связи между данными (моделями) и ПМ устанавливаются системными администраторами в процессе регистрации ПМ в PDM–системе, вне процесса проектирования [5–11].

Постановка задачи

Задачей исследований является анализ методов и средств СУБД и, в частности, ОСУБД для оценки возможности решения с их помощью проектной задачи автоматизированного формирования структур КЭМ.

Проблемы взаимодействия моделей в PDM–системах

Современные PDM–системы рассматривают модели как пассивные объекты, содержащие форматированные наборы данных. При этом обмен данными возможен только с помощью ПО, работающего под управлением пользователей (рис. 2). В результате обработка запросов может осуществляться только в последовательном (пакетном) режиме, при котором каждый запрос на получение данных может быть выполнен только после обслуживания всех предыдущих запросов. Это существенно снижает производительность исследованных PDM–систем, поскольку обмен проектными данными и их идентификация являются наиболее часто повторяющимися проектными операциями.

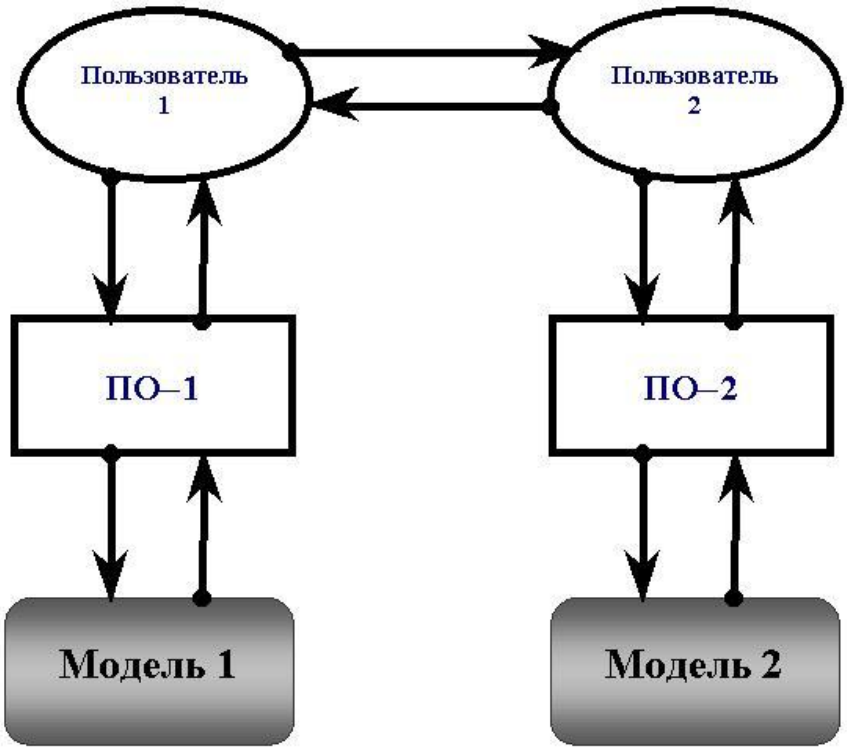

Рис. 2. Схема обмена данными в PDM – системах

ИТ автоматизированного формирования КЭМ конструкции СТО без использования ГМ может быть реализована только на основе принципа

декомпозиции задачи описания алгоритма формирования структуры модели. Такая задача не может быть решена для модели, описывающей конструкцию СТО в целом, из–за большой размерности N вектора параметров СТО $x=(x_1,x_2,...,x_N)$. Общая КЭМ конструкции СТО формируется путем последовательного объединения локальных КЭМ подконструкций и сборок (синтез КЭМ). При этом количество транзакций между моделями возрастает пропорционально количеству моделей, которое, в свою очередь, зависит от числа уровней декомпозиции. Пассивный статус моделей в PDM–системах является причиной возникновения дополнительных ошибок при обмене данными между ними, поскольку количество ошибок ПО, входящих в состав систем автоматизированного проектирования существенно зависит не только от качества разработки ПО, но также и от правильности установления связей между ПО и данными, хранящимися в БД [3]. Рассматривая модели в виде сравнительно автономных пассивных объектов, PDM–системы не обеспечивают эффективного обмена данными между моделями средствами СУБД. Таким образом PDM–системы могут управлять синтезом моделей, только если этот процесс не сопровождается обменом данными между моделями (синтез 3D–моделей).

Анализ функциональных особенностей СУБД

Основные системы PDM базируются сегодня на реляционных СУБД (РСУБД), лидирующей среди которых является продукт фирмы Oracle. Тип базовой СУБД определяет структуру и свойства ПО, которое может быть использовано PDM–системой для анализа и обработки данных, а также методы и средства управления данными.

Реляционная модель данных не допускает естественного представления данных со сложной структурой, например КЭМ, поскольку в ее рамках возможно моделирование лишь с помощью плоских отношений (таблиц). Все отношения принадлежат одному уровню, многие значимые связи между данными либо теряются, либо их поддержку приходится осуществлять в рамках конкретной прикладной программы. В составе PDM–систем РСУБД в основном обеспечивает регистрацию моделей и формирование выборок на основании заданных атрибутов Кроме того, РСУБД обеспечивает авторизацию пользователей, поддержку функции электронных подписей и т.п.

Реляционная модель данных не предусматривает инкапсуляции данных на уровне концептуальной модели, используемой администраторами данных. В результате существенно затрудняется обмен данными между локальными информационными системами, входящими в состав САПР. По определению в реляционной модели поля кортежа могут

содержать лишь атомарные значения, в то время как САПР оперируют со сложно−структурированными объектами [12]. Даже в том случае, когда сложный объект, например КЭМ, удается спроецировать на реляционную модель данных, его данные распределяются по многим таблицам. Обмен информацией с реляционной моделью данных осуществляется на основании стандартных SQL−запросов, что противоречит основному принципу проектирования, так как подразумевает использование готовых данных. Кроме того, структуры данных, получаемые в результате запросов, имеют характер отношений, а следовательно не могут непосредственно использоваться прикладным ПО, с которым работают PDM−системы.

Единственным типом современных СУБД, в которых реализован принцип инкапсуляции данных, являются ОСУБД. В результате исследований ОСУБД IBM Lotus Notes/Domino, Jasmine, ObjectStore, Cache, **Cerebrum, db4objects,** Objectivity 5.0, ONTOS DB 2.5, Versant, Release 5, Gemstone 5.0, POET 5.0, O2 5.0, Itasca 5.0, UniSQL 3.2 и ODB−Jupiter 2.1 было установлено, что ОСУБД обеспечивают более оптимальное решение задач управления проектными данными, поскольку объекты можно хранить и использовать непосредственно, не раскладывая их по таблицам [13−24]. Кроме того, типы данных определяются разработчиком и не ограничены набором предопределенных типов. Данные и методы объектов помещаются в хранилище как единое целое, чем, в частности, обеспечивается возможность инкапсуляции.

При этом ОСУБД реализуют весь набор функций, традиционно присущих системам управления базами данных. Объектные базы данных обеспечивают доступ к различным источникам данных, в том числе к данным реляционных СУБД, а также предоставляют разнообразные средства манипуляции с объектами баз данных.

В своих программах разработчики используют объекты и структуры, которые помещаются в базу данных. Типовая схема функционирования исследованных ОСУБД приведена на рис. 3.

При этом ОСУБД поддерживают ряд специфических функций управления доступом и защиты данных, характерных только для объектно−ориентированных ИТ:

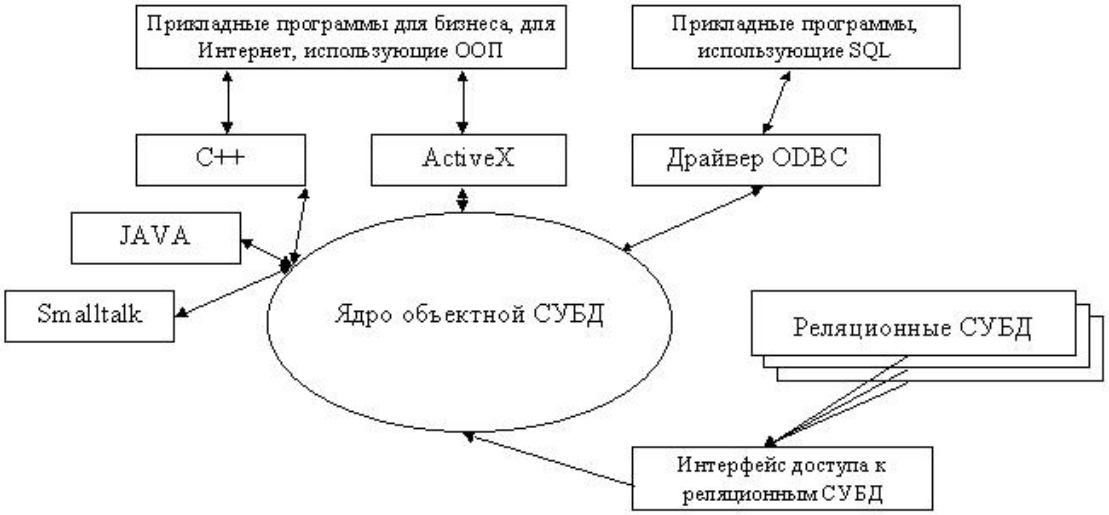

Рис. 3. Типовая схема функционирования ООСУБД

- **наследование, инкапсуляция данных и полиморфизм**;
- **идентификация объектов** – присвоение объектам идентификаторов, обеспечивающих уникальность каждого объекта в СУБД; обычно идентификатор невидим, о нем "знает" только СУБД, но он не изменяется в течение всего времени существование объекта;
- **целостность данных** – обеспечение структурной и логической целостности БД, а также поддержание соответствия между объектами в БД и их копиями, переданными приложениям;
- **параллелизм** – механизм разрешения конфликтов, возникающих при одновременном доступе к одним и тем же данным;
- **стабильность** – свойство ОСУБД сохранять данные объектов, объявленных как стабильные, между сеансами работы программ;
- **восстанавливаемость** – реакция на сбои в прикладных программах, которые выполняют обмен данными с БД, сбои операционной системы и повреждения носителей информации и т.п.;
- **транзакции** – механизм, обеспечивающий фиксацию состояния БД на момент начала какой–либо критической операции и возвращение к этому состоянию в случае неудачного выполнения операции;
- **безопасность данных** – разделение доступа к данным, с целью обеспечения конфиденциальности и защиты от несанкционированного доступа;
- **поддержка системы запросов**;
- **версионность** – поддержка на уровне СУБД многих версий одного объекта;
- **архивирование** – использование средств программного сжатия данных.

В то же время механизм реализации перечисленных функций не может считаться оптимальным. Например, процедура обеспечения **целостности данных** является внешней по отношению к объектам, в результате чего усложняется процедура установления связей между объектами. Механизм обеспечения **параллеризма** является сравнительно сложным, поскольку связан с обменом данными между ПП и ядром ОСУБД. Гараздо проще данная проблема может быть решена ОСУБД, оперирующей объектами, для активизации которых не требуется загрузка в оперативную память ПП. Механизм обеспечения **безопасности данных** основан на традиционных ИТ управления доступом, использующих для этой цели СУБД, которая является внешней по отношению к внутриобъектному ПО. В случае использования ОСУБД, оперирующей объектами, для активизации которых не требуется загрузка в оперативную память ПП, задача обеспечения **безопасности данных** может быть эффективно решена с помощью специализированного внутриобъектного ПО. Кроме того механизм обеспечения **версионности** в такой СУБД может быть гараздо проще реализован путем генерации субобъектов [25].

Анализ схемы, приведенной на рис. 3, показывает, что в соответствии с концепцией исследованных ОСУБД хранящиеся в БД объекты, а значит и хранящиеся в них модели, сами по себе не активны, поскольку их методы могут быть активизированы только в оперативной памяти прикладных программ. Например, ОСУБД ObjectStore обеспечивает долговременное хранение в базе данных объектов, созданных программами на языках C++ и Java. Вся работа с объектами ведется обычными средствами включающего ОО–языка. При этом СУБД как бы расширяет виртуальную память операционной системы. Происходит это следующим образом. Когда прикладная программа обращается к объекту, то ищет его по адресу в оперативной памяти. Нужная страница оперативной памяти может быть вытеснена в виртуальную память (область хранения неиспользуемых страниц оперативной памяти на диске). Если объекта с таким адресом в виртуальной памяти не существует, то операционная система генерирует ошибку. СУБД эту ошибку перехватывает и извлекает объект из базы данных. При такой ИТ количество классов используемых алгоритмов фиксировано и определяется в процессе первичного кодирования, что не обеспечивает динамического подключения алгоритмов.

Выводы

Проведенный анализ показал, что концепция исследованных ОСУБД не обеспечивает решения задачи объединения (синтеза) структур КЭМ. Это связано с невозможностью динамического подключения алгоритмов, поскольку перечень классов объектов, содержащихся в БД определяется

перечнем классов, на которые имеются ссылки в исходных текстах прикладных программ.

Для обеспечения синтеза структур КЭМ наряду с динамическим подключением алгоритмов ОСУБД должны также поддерживать эффективную технологию обмена данными произвольного формата и объема.

Список использованной литературы

1. *Шимкович Д.Г.* Расчет конструкций в MSC/NASTRAN for Windows // М: ДМК Пресс, 2001. –448 с.
2. *К.А.Басов.* ANSYS в примерах и задачах // М.: КомпьютерПресс, 2002. – 224 с.
3. *Борисов В.В.* Проблемы обеспечения надежности функционирования PDM–систем. // Збірник наукових праць. Технології створення перспективних комп'ютерних засобів та систем з використанням новітньої елементної бази. НАН України Інститут кібернетики ім. В. М. Глушкова, 2000. – С. 67 – 72.
4. *Майерс Г.* Надежность программного обеспечения. М: Мир, 1980. – 353 с.
5. Интегральная система разработки изделия PDS. - http: // www/ urss.ru/.
6. Windchill, ProjectLink. - http: // www/ urss.ru/.
7. Windchill, как средство организации и ведения технических архивов. - http: // www/urss.ru/.
8. Программный комплекс АРМ WinMachine. –http://www.consistent.ru/download/marketing.
9. Разговор о возможностях системы Unigrphics. –http://www.consistent.ru/download/marketing.
10. CATIA, универсальная CAD/CAM/CAE/PDM система. –http://www.catia.ru/productsNEW.htm
11. Autodesk Inventor Professional. –http://www.consistent.ru/download/marketing.
12. Ограничения реляционных баз данных: –http://www.mstu.edu.ru/education/materials/zelenkov/ch_6_1.html
13. –http://www.idlab.net
14. Обзор объектно–ориентированной СУБД Jasmine: –http://www.citforum.ru/seminars/cbd2001/day_2_1_jasmin.shtml
15. Inter Systems CACHE –http://lusindane.at.tut.by/files/index.html#top
16. –http://www.shuklin.com/ai/ht/ru/cerebrum
17. db4Objects: –http://www.db4o.com/about/productinformation/db4o

18. ONTOS DB: –http://www.inteltec.ru/publish/articles/objtech/oodbms_o.shtml
19. –http://en.wikipedia.org/wiki/Versant
20. –http://www.math.rsu.ru/smalltalk/obzornew–3.ru.html
21. –http://www.citforum.ru/database/articles/subd_cis.shtml
22. –http://osp.aanet.ru/pcworld/1998/04/74.htm
23. Создание среды хранения объектов над иерархической СУБД. –http://www.cognitive.ru/innovation/sbornic1/poray.doc
24. –http://fizmat.vspu.ru/citforum/seminars/cbd99/inteltec_4.shtml
25. *Борисов В.В.*, Объектная система управления данными "SPACE" // Труды IV Международной научно–технической конференции "Гиротехнологии, навигация, управление движением и конструирование авиационно–космической техники", посвященной 100–летию со дня рождения акад. С. П. Королева, НТУУ "КПИ". Киев, 2007. т. 2, –с. 55–61.