

Болотцев Дмитрий Анатольевич

аспирант

ФГБОУ ВПО «Воронежский государственный
технический университет»

г. Воронеж, Воронежская область

Успехов Андрей Александрович

генеральный директор

ООО «ИНОБИТЕК»

г. Воронеж, Воронежская область

Чижев Михаил Иванович

д-р техн. наук, профессор

ФГБОУ ВПО «Воронежский государственный
технический университет»

г. Воронеж, Воронежская область

МЕТОДЫ УСТОЙЧИВОГО ИМЕНОВАНИЯ ПРИМИТИВОВ В ПРОЦЕССАХ ОБМЕНА ТВЕРДОТЕЛЬНОЙ ГЕОМЕТРИЕЙ МЕЖДУ САПР С СОХРАНЕНИЕМ ИСТОРИИ ПОСТРОЕНИЯ

Аннотация: в статье рассмотрена проблема устойчивого именования примитивов, возникающая при реализации макропараметрического метода обмена моделями в САПР. Приведён краткий обзор существующих решений. Предложены методы устойчивого именования, проведено сравнение производительности предложенных методов на примерах реальных моделей.

Ключевые слова: САПР, обмен моделями, макропараметрический подход, устойчивое именование.

На сегодняшний день существует большое количество САПР, которые обеспечивают автоматизацию проектно-конструкторских и технологических работ (CAD/CAM/CAE). По мере увеличения требований к проектируемым изделиям, на этапах разработки возникает необходимость использования множества САПР,

отвечающих тем или иным целям проектирования. Во многих случаях проектирование не ограничивается применением одной САПР. Также использование только одной САПР сильно затруднено в условиях кооперативной работы над изделием, когда проектные подразделения разделены территориально и юридически, а также ограничены возможностями использования САПР, единой для всего проекта. В настоящее время для указанных процессов характерна одновременная эксплуатация множества САПР [9].

В современных промышленных условиях цифровая модель разрабатываемого изделия хранится централизованно в формате той исходной САПР, где она разрабатывается. Часто во время проектирования требуется трансляция цифровой модели в другие САПР, ориентированные на решение определенного спектра задач. После расчета данные решения используются для модификации модели по мере необходимости. Однако, эти модификации сейчас могут происходить только в исходной САПР путем изменения соответствующих параметров.

Характерный пример проблемы обмена данными возникает, когда сложная цифровая модель изделия передается из CAD системы для моделирования в тяжелую CAE систему с целью оценки её прочностных, температурных или других свойств. Часто передача осуществляется неоднократно, поскольку после таких оценок требуется внесение изменений, отражающихся на всех уровнях проекта – от твердотельной модели до чертежей, сделать которые в контексте CAE системы затруднительно из-за отсутствия в импортированной цифровой модели параметрической информации.

Современные подходы к обмену данными между CAD системами предлагают:

1. Использование нейтральных (рисунок 1) форматов (IGES, STEP, DXF, JT и т. д.). Эти файлы содержат точное геометрическое описание модели (BRep модель). Конвертируемые детали импортируются в другую CAD систему в виде единого тела, поверхности или набора поверхностей (без дерева моделирования).

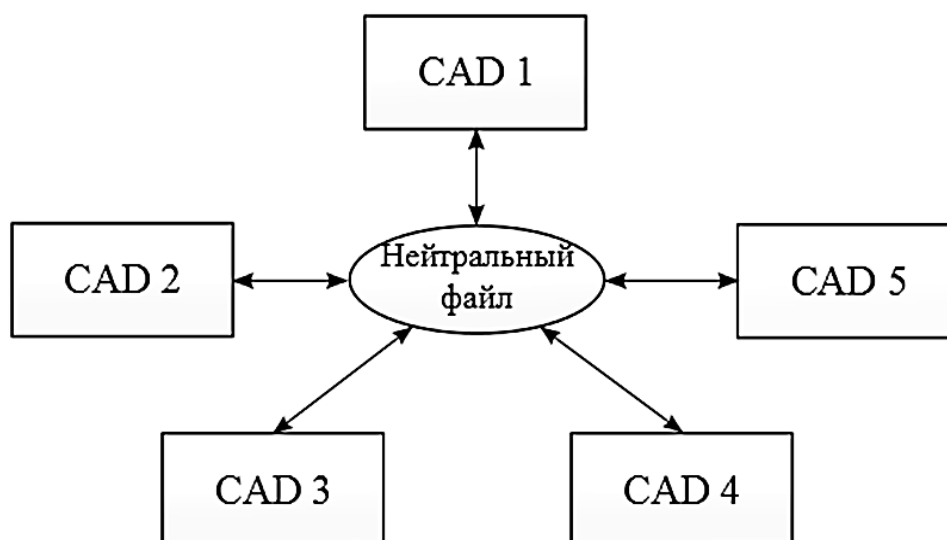


Рис. 1. Схема обмена данными между САПР при использовании нейтрального формата

2. Использование прямых трансляторов. Они позволяют передавать геометрическую информацию между САПР, не прибегая к промежуточным преобразованиям через нейтральный формат. Подход лишен недостатков первого направления, однако импортируемая геометрия также конвертируется в единое тело из поверхностей (без дерева моделирования).

К тому же для N различных CAD систем потребуется $N(N-1)$ трансляторов (рисунок 2), которым необходимо постоянное обновление и поддержка последних версий. Это является затратным как по экономическим, так и по техническим соображениям.

3. Использование макропараметрического подхода предложенного Корейским ведущим научно-техническим институтом, где вместо обмена геометрическим описанием детали происходит обмен данными о командах моделирования, которые имеют единый для всех CAD систем формат [10]. Информация из макрофайла одной CAD системы транслируется в нейтральный XML файл, который далее используется для преобразования в набор макрокоманд другой CAD системы [1].

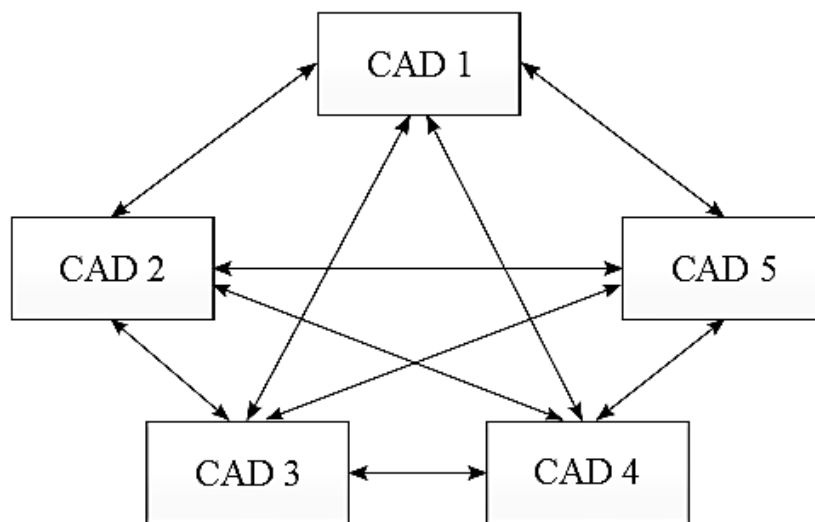


Рис. 2 Схема обмена данными между САПР при использовании прямых трансляторов

Авторам видится, что третье направление концептуально очень перспективно, поскольку в этом случае передаваемая информация содержит «замысел проектировщика» и соответственно историю моделирования. Однако, реализация этого направления на базе использования функционала макросов различных САПР практически неосуществима. Во-первых, этот механизм от системы к системе может либо вообще отсутствовать, либо его уровень не позволяет полностью транслировать информацию о построении модели (скудный язык описания). Во-вторых, требуется постоянная запись во время проектирования. В-третьих, макросы отражают, – помимо полезных (проектных), – множество бесполезных команд, которые необходимо далее фильтровать. В результате уже этих недостатков «макросный» подход имеет существенные ограничения для реализации на практике и может быть применим далеко не во всех случаях обмена данными между САПР.

Более того макросный подход имеет более серьезную, можно сказать, ключевую проблему. Она связана с устойчивым именованием примитивов. На каждом этапе проектирования детали (следуя действиям конструктора) происходит построение связей между геометрическими элементами цифровой модели, которые устанавливаются вручную с помощью мыши посредством выделения требуемых примитивов на экране. К примеру, твердотельный элемент «вытяжка»

определяется эскизом и направлением вытяжки. Сначала для указанного эскиза строится система координат, состоящая из примитивов, определяющих ее положение в пространстве, скажем, точки и двух векторов. Эти примитивы выделяются вручную визуальным конструктором на текущей модели, если она находится в построении или создаются отдельно. Также выделяется плоскость, в которой будет располагаться эскиз. Примитивы модели имеют идентификаторы. Таким образом, рождаются связи родитель-потомок между примитивами, элементами построения (features), а также вспомогательными объектами и каждый следующий этап построения оказывается связан с предыдущим. В итоге, для построения (воспроизведения) последовательности конструктивных элементов модели в разных САПР требуется однозначный для всех САПР механизм именования примитивов.

Данный аспект слабо освещен в современных научных кругах, однако есть соответствующие работы, ориентированные на использование данных механизмов в геометрических ядрах твердотельного моделирования.

Kripas разработал метод поиска имён с использованием графа идентификаторов граней (FaceIdGraph), который обновляется каждый раз при изменении топологии модели [4].

Caroyleas предложил метод именования с использованием топологии. Этот метод использует специфическую для feature информацию, например, профиль и траекторию в случае операции выдавливания [5].

Основываясь на методе Caroyleas, Chen разработал алгоритм поиска вершин, рёбер и граней [6].

Agbodaп предложил метод именования и метод поиска имён с использованием графа оболочек (Shell). Эти методы похожи на метод Kripas, но имеют несколько уровней детализации. Это реализовано за счёт введения иерархической архитектуры графа [2].

Mun и Nan разделили проблему устойчивого именования на две подпроблемы: проблема идентификации топологических сущностей и проблема соответствия имён в разных САПР [3].

Все перечисленные выше методы работают на уровне геометрического ядра САПР. Другими словами, назначение идентификатов примитивам происходит на этапах построения геометрии топологическим аппаратом ядра. Информация о стратегии именования не стандартна и является объектом авторских прав той или иной компании-производителя геометрического ядра, вследствие чего невозможно достичь одинакового именования примитивов геометрически эквивалентных моделей в разных САПР.

Проблемы устойчивого именования привели развитие макропараметрического подхода к использованию API САПР. Это было логично продиктовано существенными недостатками макросов и широкими возможностями доступа к геометрической информации с помощью API. В статьях [8; 7] авторами предлагается развитие макропараметрического подхода в сторону использования API конкретной САПР. В этом подходе вводится новый уровень управления САПР через API. Вместо XML для описания нейтральных инструкций построения используется простой и мощный язык программирования JavaScript. API подход инвариантен к использованию языка инструкций; это может быть произвольный подходящий для использования язык программирования и даже XML в специальном «программном» формате описания. Введение языка программирования обусловлено предоставлением человеку возможности ручного кодирования стандартных операций построения параметрической конструкторской базы данных стандартных деталей и их составных элементов. Были выполнены следующие задачи: разработан короткий набор нейтральных JavaScript инструкций построения модели на основе ранее выработанного списка команд; реализована простейшая функция «раскрутки» дерева построения (препроцессирование) и генерации сценария построения на нейтральном языке моделирования; реализован алгоритм восстановления модели по полученному сценарию (постпроцессирование). Использовалась идентификация примитивов по методу «сферы». Этот метод возникает из естественных действий инженера-проектировщика и, по сути, является эмуляцией действий мышью. Выделение объекта происходит с помо-

щью точки, для которой задаются координаты. Далее, перебором всех топологических примитивов подбирается такой, кратчайшее расстояние от характерной точки которого до заданной точки минимально. Если наименьшее расстояние от характерной точки топологического примитива до заданной точки больше порогового значения, то считается, что заданная окрестность точки не содержит никаких примитивов. Если есть несколько примитивов с одинаковым расстоянием до заданной точки, то предпочтение отдаётся сначала вершине, затем ребру, а после – грани.

В продолжение развития подходов к устойчивому именованию предлагается усовершенствованный вариант предыдущей методики, метод, использующий октаэдрное дерево (октодерево) для поиска примитива и новый способ «сквозного» именования примитивов.

Усовершенствование алгоритма «сферы» заключается в ведении фильтра примитивов. Теперь при выполнении поиска топологического элемента можно указать тип примитива (вершина, ребро или грань), который следует выбрать.

Согласно данным в таблице 1, на поиск примитива по улучшенному алгоритму «сферы» требуется примерно в 3 раза меньше времени. Модель, на которой проводилось тестирование изображена на рисунке 3. Точность 0,01 мм, радиус сферы 0,01 мм.

Таблица 1

Сравнение скорости поиска примитивов по старому
и улучшенному алгоритму «сферы»

<i>Тип искомого примитива</i>	<i>Время поиска по алгоритму «сферы», мс</i>	<i>Время поиска по улучшенному алгоритму «сферы», мс</i>
Грань	981	356
Ребро	796	92
Вершина	778	374

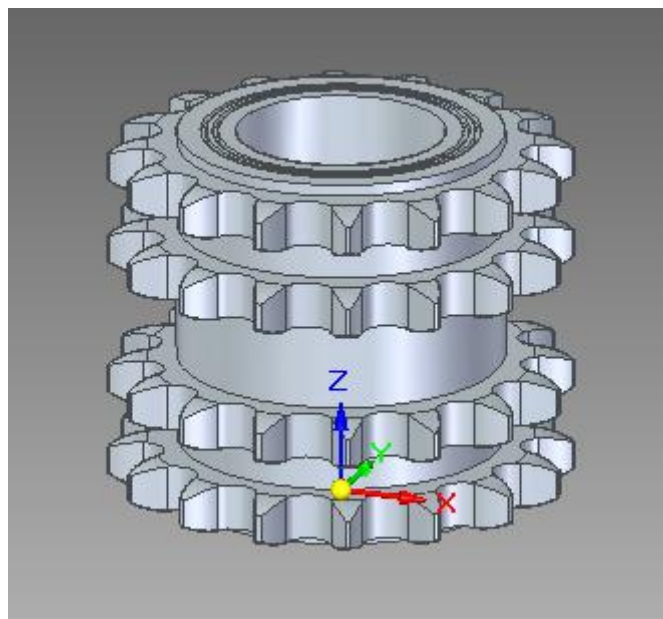


Рис. 3. Тестовая модель

Алгоритм сквозного именования основан на пространственной идентификации примитива в пространстве. В примитивах можно выделить характерную точку, в которой условно сконцентрирована вся геометрия каждого из них.

Для вершин координаты характерной точки совпадают с координатами самой вершины.

Для рёбер характерная точка – это такая точка, в которой параметр t кривой, описывающей геометрию ребра, принимает среднее значение:

$$t_{cp} = |t_{max} - t_{min}| / 2, \quad (1)$$

где t_{max} и t_{min} – это значения параметра t на концах ребра, $t = f(x, y, z)$.

Данное определение характерной точки не подходит для тех случаев, когда ребро представляет собой замкнутый контур. Для таких контуров не в чистом представлении не существует топологических вершин. API многих современных САПР позволяют получить минимальное и максимальное значения параметра t для рёбер, но так как в разных САПР используются различные математические аппараты, определить характерную точку такого контура однозначно невозможно. Поэтому, для замкнутых рёбер-контуров было предложено другое определение характерной точки – это точка с «минимальными» координатами. Находится она следующим образом: вычисляется точка, лежащая на контуре, с

наименьшей координатой по X. Если она – единственная, то такая точка становится характерной. Если таких точек – множество (контур, лежащий в плоскости ZY), вычисляется точка, лежащая на контуре, с наименьшей координатой по Y и эта точка становится характерной.

Для граней координаты характерной точки совпадают с координатами точки, в которой параметры u , v принимают своё среднее значение:

$$u_{cp} = |u_{max} - u_{min}| / 2, \quad (2)$$

$$v_{cp} = |v_{max} - v_{min}| / 2, \quad (3)$$

где u_{max} и u_{min} максимальное и минимальное значения параметра u , v_{max} и v_{min} максимальное и минимальное значения параметра v ,

$$u, v = f(x, y, z). \quad (4)$$

Для каждого типа примитивов создаётся массив характерных точек, которые сортируются по возрастанию координат X, Y, Z. Далее, примитивам выдаются имена в виде Ni , где N – это вид примитива («v» для вершин, «e» для рёбер, «f» для граней), i – порядковый номер соответствующей характерной точки в отсортированном массиве. Пара «имя» и «примитив» записываются в карту имён.

Чтобы визуализировать результат работы алгоритма сквозного именования, топология модели с именами примитивов выводится в файл, который открывается в программе трёхмерного моделирования Blender в виде каркасной модели. Примеры работы алгоритма на рис 4. Время поиска примитива по имени в карте имён очень мало, однако, время, необходимое на заполнение карты имён примерно в 4,5 раза больше времени, необходимого для выполнения поиска трёх примитивов по улучшенному методу «сферы» (таблица 2). Кроме того, после каждого изменения модели следует заново составлять карту имён примитивов.

Таблица 2

Сравнение времени поиска примитивов по улучшенному алгоритму «сферы» и времени построения карты имён

<i>Тип искомого примитива</i>	<i>Время поиска по улучшенному алгоритму «сферы», мс</i>	<i>Время составления карты имён, мс</i>
Грань	8	54

Ребро	4	54
Вершина	3	54

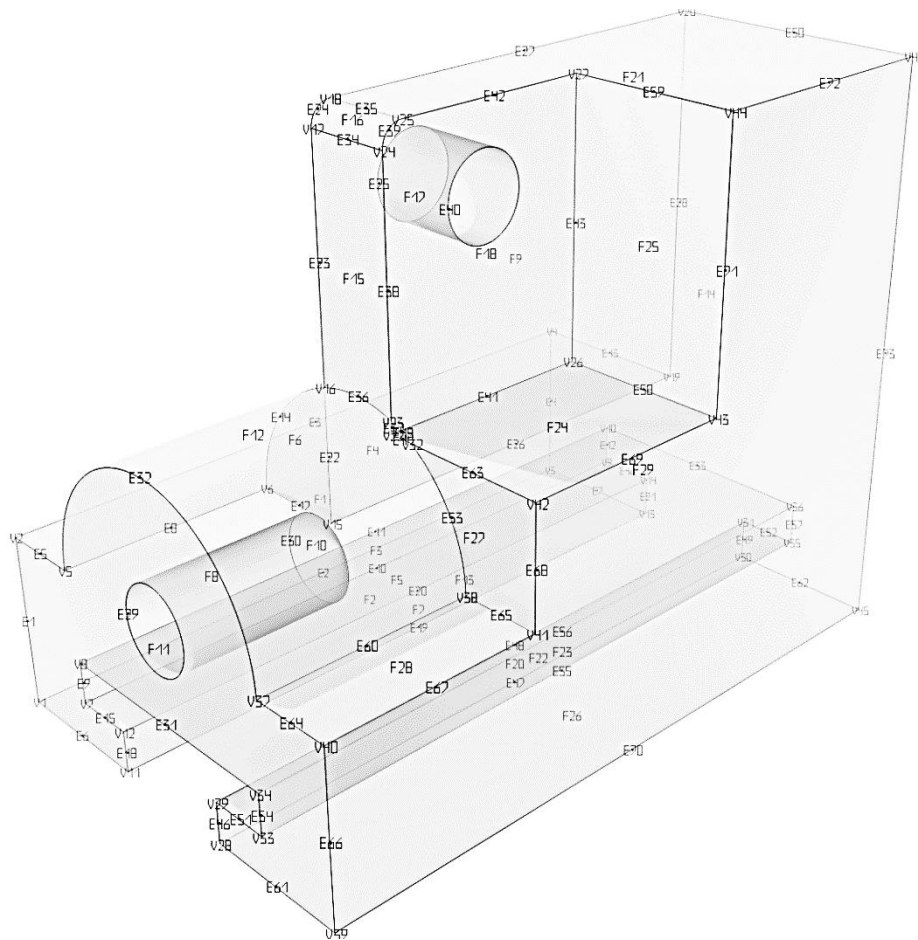


Рис. 4. Результат работы алгоритма сквозного именования

Алгоритм поиска примитива в октодереве так же использует характерные точки примитивов. Октодерев (англ. octree) – древовидная структура данных, в которой у каждого внутреннего узла ровно восемь потомков. Алгоритмы, основанные на использовании октодеревы используются для пространственной индексации, эффективного обнаружения столкновений в трёхмерном пространстве, для реализации метода конечных элементов и т. д.

Этот алгоритм состоит из двух частей: построение октодеревы и, собственно, поиск в октодереве. Построение октодеревы выполняется следующим образом: первый узел дерева является ограничивающим параллелепипедом (bound box), который содержит всю модель целиком. Грани этого параллелепипеда параллельны осям OX, OY и OZ. Этот узел разбивается на 8 одинаковых

октантов. Далее, вычисляется количество примитивов, которое попадает в каждый октант. Если в октанте более одного примитива, то он разбивается на подоктанты, для каждого из них выполняется проверка количества примитивов и так далее до тех пор, пока в каждом октанте не останется 1 или 0 примитивов.

После того, как будет построено октодерево, можно выполнять поиск примитива. Для точки в пространстве, заданной координатами x, y, z проверяем, попала ли точка в октант. Если нет, то завершаем поиск в этом октанте. Если же точка попала в октант, то для каждого из его дочерних октантов выполняем проверку, попала ли точка в него. Так до тех пор, пока мы не придём в октант, у которого нет потомков. Результат поиска – id примитива, характерная точка которого попала в конечный октант.

В таблице 3 представлены результаты измерения скорости поиска примитива в сравнении со скоростью поиска примитива по усовершенствованному методу «сферы». Точность 0,001 мм, радиус сферы 0,001 мм. Тестовая модель на рис. 5. Согласно данным из таблицы 3, при выполнении поиска примитива в октодереве больше всего времени тратится на построение примитива. Однако не обязательно перестраивать октодерево, если между командами поиска в октодереве модель не изменяется. Тем не менее, даже без перестроения октодерева, время, затраченное на поиск примитива в октодереве, примерно в 1,5 больше времени, затраченного на поиск примитивов по улучшенному методу «сферы».

Таблица 3

Сравнение времени поиска примитива по методу «сферы» с фильтром
и времени поиска примитива в октодереве

<i>Тип искомого примитива</i>	<i>Время поиска по улучшенному алгоритму «сферы», мс</i>	<i>Время построения октодерева, мс</i>	<i>Время поиска примитива в октодереве, мс</i>	<i>Общее время поиска примитива, мс</i>
Грань	285	624	0	624
Ребро	26	624	0	624
Вершина	103	624	0	624

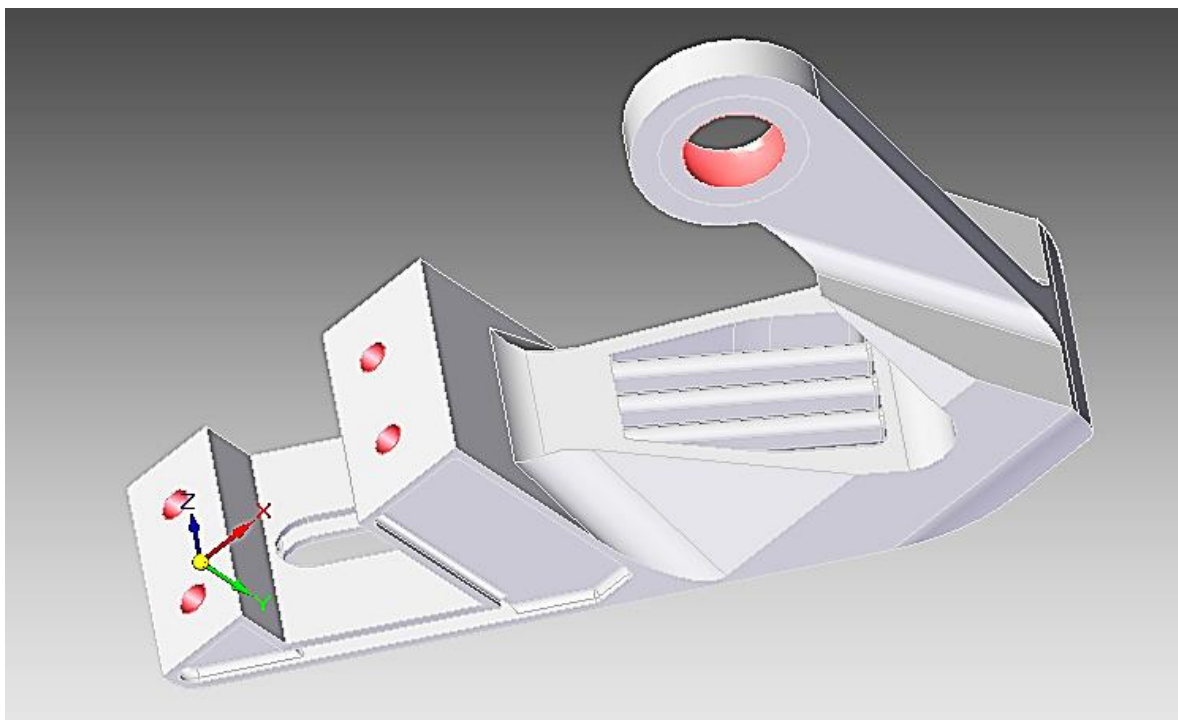


Рис. 5. Тестовая модель

Выводы

1. Существующие методы решения проблемы устойчивого именования примитивов привязаны к конкретным геометрическим ядрам, вследствие чего невозможно достичь одинакового именования примитивов геометрически эквивалентных моделей в разных САПР.

2. Авторами статьи были разработаны алгоритм поиска примитивов по методу «сферы», метод «сквозного» именования, алгоритм поиска примитивов в октодереве.

3. Использование фильтра по типу примитива (грань, ребро, вершина) в методе «сферы» позволяет сократить время поиска примитива примерно в 3 раза.

4. Алгоритм «сквозного» именования требует временных затрат на построение карты имён, однако обеспечивает почти мгновенный поиск примитива по имени.

5. При использовании алгоритма поиска примитива в октодереве, время, требующееся на построение октодереве соизмеримо с тремя-четырьмя вызовами метода поиска примитива по алгоритму «сферы».

6. Методы «сферы» и поиска примитива в октодереве отражают действия инженера-проектировщика, поэтому их можно использовать при ручном программировании. Однако за одно выполнение алгоритма «сферы» можно выбрать не более одного примитива. Поскольку для построения геометрии на существующей модели требуется три примитива (например, для построения системы координат нужно выбрать плоскость, прямую и точку), требуется три вызова метода «сферы», тогда как после одного построения октодереве до изменения модели можно практически мгновенно получать примитивы, используя алгоритм поиска примитива в октодереве.

Список литературы

1. Choi G. Exchange of CAD part models based on the macro-parametric approach / G. Choi, D. Mun, S. Han // International Journal of CAD/CAM. – 2002. – Vol. 2. – №1. – P. 23–31.
2. D. Agbodan, D. Marcheix and G. Pierra. Persistent naming for parametric models // International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision. – 2000.
3. D. Mun, S. Han. Identification of Topological Entities and Naming Mapping for Parametric CAD Model Exchanges // International Journal of CAD/CAM. – 2005. – Vol 5. – №1.
4. J. Kripac. A mechanism for persistently naming topological entities in history-based parametric solid models // Computer-Aided Design. –1997. – Vol. 29. – №2. – P. 113–122.
5. V. Capoyleas, X. Chen, C. M. Hoffmann. Generic naming in generative, constraint-based design // Computer-Aided Design. –1996. – Vol. 28. – №1. – P. 17–26.
6. X. Chen, C. Hoffmann, «On editability of feature-based design» // Computer-Aided Design. – 1995. – Vol. 27. – №12. – P. 905–914.
7. Чижев М.И. Перенос параметрической модели между САПР»: Материалы XI международна научна практична конференция «Найновите Научни Постижения – 2015» (София, 17–25 март 2015) / М.И. Чижев. – Т. 14. – С. 45–48.

8. Чижов М.И. Построение элементарных операций твердотельного моделирования на базе уникальных в САПР: Materiały X Międzynarodowej naukowo-praktycznej konferencji «Wykształcenie i nauka bez granic – 2014» (Przemyśl, 07 – 15 grudnia 2014 roku). Matematyka. Fizyka.: Przemyśl. Nauka i studia. – 2014. – Vol. 25. – С. 89 – 91.

9. Электронный каталог Isicad – Российский информационно-аналитический портал, публикующий мировые и отечественные новости САПР, PLM и ERP [Электронный ресурс]. – Режим доступа: http://isicad.ru/ru/articles.php?article_num=15776

10. Электронный каталог Macro-parametrics Approach [Электронный ресурс]. – Режим доступа: <http://www.macro-parametrics.org/>