

Уже многие годы в автоматизированном машиностроительном проектировании (Mechanical CAD) доминируют технологии моделирования, основанные на фиксации всех действий в иерархическом дереве построений (часто такое моделирование называют параметрическим, что не совсем верно). Двадцать лет назад на рынок вышел основоположник этого направления – Pro/ENGINEER, затем появились и стали успешно развиваться такие пакеты, как SolidWorks, Solid Edge, Inventor и др. Несмотря на всю их популярность, в последнее время оживился интерес к альтернативным подходам. Всё чаще мы слышим о “чистой геометрии”, “прямом редактировании” и пр.

Предлагаемый вниманию читателей подробный обзор позволит глубже разобраться в базовых технологиях, применяемых в MCAD-системах, уточнить понятия и терминологию.

Азбука технологий моделирования в MCAD-системах

Часть II. Намерения проекта не обязательно должны быть видны

(Продолжение. Начало в #6/2007)

Paul Hamilton (p.hamilton@phusionengineering.com)

©2007 Ash Bridge Media LLC

Paul Hamilton – президент и CEO американской компании PHusion Engineering Solutions LLC (www.phusionengineering.com), которая специализируется на повышении инновационного потенциала и продуктивности компаний-клиентов, помогая им в выборе, внедрении и интеграции CAD-, PDM- и PLM-инструментов и технологий, соответствующих направлению развития их бизнеса.

Ключевое различие CAD-технологий с историей построений и без нее (напомню, что последнюю теперь иногда называют динамическим моделированием или технологией прямого редактирования геометрии модели) состоит в том, как фиксируется и как затем используется **design intent** – совокупность сведений, образующих замысел конструкции и отражающих намерения проектировщика. В контексте автоматизированного трехмерного машиностроительного проектирования под термином **design intent** обычно понимается совокупность параметров, ограничений, размеров, взаимосвязей и конструктивных элементов, которые дополняют 3D-модель.

Структура CSG-дерева (Constructive Solid Geometry – о конструктивной блочной геометрии см. в первой части статьи в #6/2007. – Прим. ред.) в гибридных (Hybrid) моделлерах облегчает фиксацию и хранение сведений о геометрии в дереве построений. Эта информация необходима, чтобы CSG- или Hybrid-моделлер мог работать. Впечатляющая маркетинговая деятельность отдельных CAD-компаний заставила нас поверить, что только гибридное моделирование с историей построений является единственным способом привнести в 3D-геометрию интеллектуальность, добавив имеющиеся у автора разработки сведения о создаваемой конструкции (intelligence), причем таким образом, чтобы это позволило (или, наоборот, не позволило) проводить дальнейшие модификации, соответствующие авторскому замыслу. На самом деле, CSG- и Hybrid-моделлеры просто не могут работать без таких сведений. Нельзя сказать, что Hybrid-технология является единственной, с помощью которой их можно добавить к геометрии модели. Этот

метод является простейшим, однако положение дел здесь быстро меняется.

Каким образом и когда сведения о конструкции (intelligence) добавляются к модели – это отдельный вопрос. Для пользователей, работающих с деревом построений, введение такой информации является одним из шагов в процессе моделирования. Благодаря этому модель может быть корректно воссоздана после её модификации. Некоторые моделлеры не дают возможность завершить процесс моделирования до тех пор, пока везде не заданы необходимые взаимосвязи и параметры. В других системах пользователям необходимо позднее вернуться в дерево построений и задать требуемую информацию – такой подход к моделированию называют вариационным (variational). Во всех этих случаях взаимосвязи определяются последовательностью шагов моделирования (историей построений).

Для систем без истории построений, опирающихся на граничное представление (B-Rep), уже сейчас есть технологии (и они постоянно развиваются), позволяющие в любое время в процессе проектирования добавить к детали или сборочному узлу сведения о намерениях проектировщика (design intent). Поскольку эту интеллектуальность нельзя встроить в дерево построений (ввиду его отсутствия), CAD-система нуждается в параметрическом солвере, с помощью которого можно сделать оценку всех взаимосвязей и на основе полученных результатов ввести изменения в геометрию модели. Такой солвер оценивает всю модель целиком, а не последовательно по шагам, как это делается при движении по дереву построений. Примером такой технологии может служить система D-Cubed от Siemens PLM Software. Поскольку

взаимосвязи “цепляются” напрямую к геометрии, а не к дереву построений или к элементам этого дерева, то источник или способ создания геометрии могут быть любыми, включая файлы в формате *IGES* или *STEP*.

Как уже отмечалось, компания *CoCreate Software* (перешедшая ныне в собственность *PTC*. – *Прим. ред.*) предлагает на рынке параметрический солвер для систем без истории построений, который, по-видимому, обладает наиболее развитыми возможностями, в числе которых и управляемые уравнениями связи. Одной из самых значительных проблем этой методологии является необходимость “убедить” пользователя отвести время для добавления в модель интеллектуальности, поскольку такая процедура при разработке или модификации модели не является принудительной, как это имеет место в процессе *CSG*- или *Hybrid*-моделирования. Несмотря на то, что такое добавление является несколько более сложным делом, эта технология уже работает.

Конструктивные элементы

Важной структурной единицей в *CSG*- и *Hybrid*-моделлерах является конструктивный элемент (*feature*). Этот термин допускает много толкований, которые могут ввести в заблуждение. Если конструктивный элемент (**КЭ**) рассматривать в контексте *Hybrid*-моделирования, в большинстве случаев он представляет собой примитив в дереве построений. Есть также некоторые исключения – скругления (*rounds*) и сопряжения (*blends*). (Фактически это сложные поверхности, получаемые путем интерполяции или усреднения ступенчатой границы между двумя поверхностями. – *Прим. ред.*.)

К примеру, чтобы с помощью *Hybrid*-моделлера проделать отверстие в теле, имеющем форму прямоугольного параллелепипеда (*block*), необходимо выполнить следующие операции. Сначала надо выбрать базовый примитив нужной формы, и разместить его в *3D*, потом добавить цилиндр соответствующего размера в некотором пространстве. Затем нужно выполнить операцию булева вычитания цилиндра из прямоугольного параллелепипеда. (Эти шаги проиллюстрированы в следующем параграфе.)

Цилиндр является дочерним или порожденным элементом (*child*) и может быть связан с параллелепипедом (родительским или порождающим элементом – *parent*) различными способами. Основное следствие такой взаимосвязи заключается в том, что определенные поверхности (в зависимости от того, как модель создана) образуют группу и будут оставаться сгруппированными в течение всего времени жизни модели. Взаимосвязь поверхностей (конструктивных элементов) будет оказывать существенное влияние на последующие трансформации модели. В целом, чем более сложным является конструктивный элемент/примитив (то есть, чем больше

он имеет поверхностей с наложенными взаимосвязями), тем меньшая степень гибкости будет возможна при будущих модификациях.

В системах без истории построений КЭ (примитивы) не создаются на каждом шаге моделирования геометрии и автоматически не поддерживаются. Здесь “конструктивный элемент” может быть в любой момент времени определен пользователем путем объединения в группу нескольких поверхностей. Этим группам поверхностей можно присвоить обозначения, и всё это сохраняется вместе с моделью. В любой момент групповой КЭ можно редактировать, чтобы актуализировать в соответствии с намерением проекта по мере его созревания. В этом контексте конструктивные элементы приобретают существенно большую значимость для проекта в отличие от ситуаций, когда они являются просто результатом выполнения шагов моделирования в процессе создания модели.

Структура “родитель – потомок”

Другой важный тип взаимоотношений между элементами в *Hybrid*-моделлерах – это взаимоотношения типа “родитель – потомок”. В таких *CAD*-системах невозможно избежать появления этих отношений, поэтому надо научиться управляться с ними корректно.

У взаимоотношений типа “родитель – потомок” имеется ряд преимуществ, которые не смогут получить системы без истории построений. В качестве примера рассмотрим конусообразный выступ (*tapered boss*). Интерес в этом случае обычно представляет диаметр конуса и его угол. Для *CSG*- или *Hybrid*-моделирования конус является элементом, порожденным цилиндром. Можно просто вернуться в дерево построений и изменить диаметр цилиндра, а затем регенерировать дочерний элемент, получив в результате конический выступ других размеров.

Без наличия взаимоотношений “родитель – потомок” вернуться к диаметру основной окружности и внести коррекции очень сложно. Для обработки этой ситуации разработаны специальные методы, однако этот тип изменений, являющийся естественным для *Hybrid*-моделирования, вызывает сложности в моделлерах без истории построений.

И последнее, о чём необходимо упомянуть в контексте рассмотрения геометрических взаимосвязей. Требования к ним могут быть самыми различными в зависимости от вашего бизнеса, от существующего процесса разработки изделия, от жизненного цикла изделия. В одном случае важно, чтобы взаимосвязи оставались неизменными при изменении геометрии (например, для семейства деталей), в другом случае важной является возможность изменять взаимосвязи при неизменной геометрии – система с историей построений позволяет с этим справиться. Иногда пользователи таких систем бывают вынуждены пере моделировать деталь только по одной

причине – встроенные взаимоотношения больше не удовлетворяют концепции конструкции (*design intent*).

По моему мнению, введенная в модель интеллектуальность важна. Однако является ли дерево построений наилучшим способом для этого? Только до тех пор, пока *Hybrid*-моделирование позволяет это сделать проще. А так будет не всегда.

То, что вы видите, не всегда есть то, что вы имеете?

Как сказано выше, одним из наиболее существенных отличий систем с историей и без истории построений являются взаимоотношения между элементами типа “родитель – потомок”. Благодаря наличию дерева построений пользователь взаимодействует не напрямую с геометрией модели, а с заданными в дереве построений взаимосвязями между элементами. Геометрия модели является только результатом “проигрывания” дерева построений. То есть, управляет всем дерево, а не геометрия. Другими словами, “то, что вы видите, не всегда есть то, что вы имеете” (*What You See Is not always What You Get* – противопоставление известному принципу построения пользовательского интерфейса *What You See Is What You Get*, или *WYSIWYG*. – *Прим. ред.*).

Теперь мы рассмотрим простой пример, который продемонстрирует этот тезис. Обратите внимание на последовательность шагов моделирования. Помните, что *Hybrid*-система фиксирует каждый шаг создания модели. Попробуйте самостоятельно проделать описанные шаги в любой из представленных на рынке систем с историей построений. В этом аспекте все они работают одинаково.

Итак, мы будем создавать куб, как показано на **рис. 1** (слева). Поместите эскиз квадрата, как того требует используемая вами *CAD*-система, на эскизную (*sketch plane*) или базовую поверхность (*datum plane*). Вытяните эскиз (*extrude*) на необходимую высоту. Это первый примитив и первый шаг в дереве построений, обычно называемый базовым конструктивным элементом (*base feature*). Базовый КЭ имеет следующую структуру: куб является дочерним

элементом по отношению к квадрату, а квадрат (в большинстве случаев) является дочерним элементом по отношению к базовой поверхности. Это необходимо помнить в процессе создания модели.

Следующим шагом сделаем в кубе паз. Обычно это происходит так: с гранью куба совмещается эскизная поверхность, где рисуется эскиз вырезаемого прямоугольника. С помощью этого эскиза от базового конструктивного элемента можно отсечь лишнее (команды *cut*, *mill* или *slot* – в зависимости от системы). Обычно для выполнения операции *cut* на базе эскиза создается *3D*-примитив, затем с помощью булева вычитания (*Boolean Subtract*) базовый конструктивный элемент трансформируется к виду, показанному на **рис. 1** (в центре).

Итак, какими операциями пополнилось дерево построений? Второй примитив (прямоугольник) является дочерним элементом по отношению к эскизу, эскиз является дочерним элементом по отношению к эскизной поверхности, а эскизная поверхность является дочерним элементом к той грани куба, на которую она была помещена. Теперь наша модель состоит из двух КЭ. В нашем случае термин “конструктивный элемент” действительно относится к примитивам, к которым применяются булевы операции, и которые сохраняются в дереве построений.

Третьим шагом “добавим” сквозное отверстие в боковой поверхности, как показано на **рис. 1** (справа). Если вы недостаточно продумаете свои дальнейшие действия по возможной модификации модели, то, вероятнее всего, пойдете путем наименьшего сопротивления и создадите один примитив для “отверстия через все грани”. В некоторых *CAD*-системах для отверстий имеется соответствующий КЭ, в других придется рисовать круг на поверхности; возможны и иные варианты. Итак, цилиндр, пронизывающий всё тело модели, – это примитив (КЭ), который вычитается из модели (см. **рис. 2**).

На этом работа первого конструктора в рассматриваемом нами примере завершается. Теперь модель переходит к другому, который и будет модифицировать её. Давайте посмотрим, как это происходит.

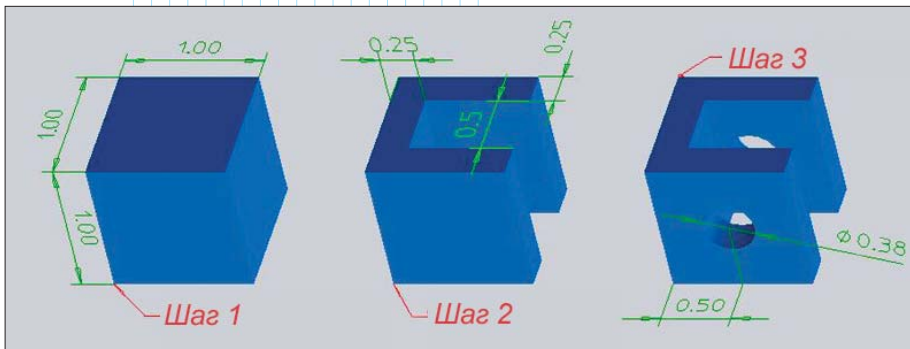


Рис. 1. Шаги создания куба с вырезом и отверстием

Модификация №1: растяжение Пластины 1

Прежде чем продолжить, важно понять, как создается модель. В *Hybrid*-системах нельзя трансформировать геометрию модели – необходимо изменить дерево построений. Геометрия изменится автоматически в соответствии с деревом построений. Здесь есть некоторое сходство с компьютерной

программой. Нельзя напрямую изменить результат работы программы – чтобы получить другой результат, надо внести коррекции в текст программы.

На рис. 2 показана готовая модель детали. Если посмотреть дерево построений, можно увидеть базовый КЭ и два других – пластину и отверстие. Можно также увидеть взаимоотношения типа “родитель – потомок” или порядок построения.

Простое созерцание изображения на экране не позволяет увидеть определение модели в полном объеме. К примеру, сколько, по вашему мнению, граней у модели? Я насчитал 12. Если вы думаете так же, вы ошибаетесь. В соответствии с деревом построений у модели имеется только 10 граней. Шесть граней – у базового примитива, еще три грани добавляет примитив паза, а еще одна поверхность – у примитива отверстия. Если сложить, то получается 10. Насколько же усложнятся подсчеты, когда вы начнете работать с моделями, содержащими сотни примитивов/КЭ!

Торцевые поверхности Пластины 1 и Пластины 2 этой модели не являются двумя отдельными поверхностями, как это может показаться. Обе они принадлежат одной грани. Эта грань появилась при создании базового конструктивного элемента; именно так она была зафиксирована в дереве построений, и такой останется в течение всего времени жизни модели – до тех пор, пока эскиз базового КЭ не будет изменен.

В этой связи необходимо быть внимательным при внесении изменений в базовый конструктивный элемент – его геометрия, как правило, определяет геометрию многих дочерних элементов. В нашем случае (в отличие от показанного на рис. 3 изображения) Пластины 1 нельзя удлинить независимо от Пластины 2. Это же относится и к отверстию. На рисунке оно выглядит как два отдельных отверстия, однако, если основываться на данных создания модели, то на самом деле – это одно отверстие. Чтобы изменить диаметр одного из изображенных на рисунке отверстий

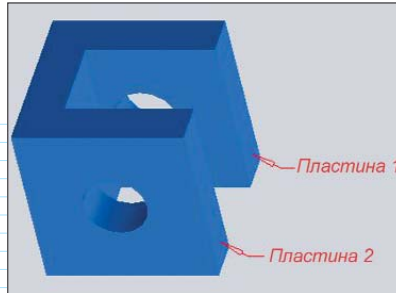


Рис. 2. Обе пластины являются составными частями одного и того же конструктивного элемента

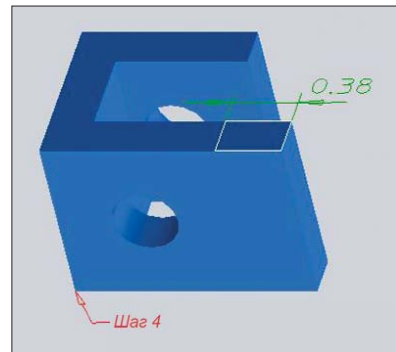


Рис. 3. Две пластины представлены одной позицией в дереве построений

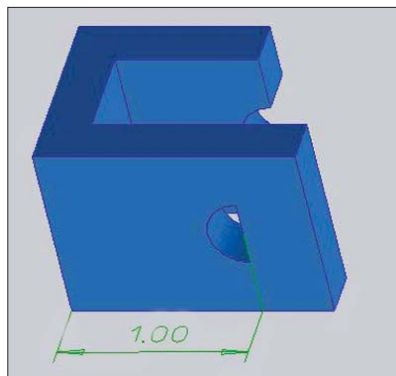


Рис. 4. Переместить отверстие не так легко, как кажется

независимо от другого, надо либо переопределить примитив отверстия и добавить еще один примитив отверстия для второй пластины, либо для одного из существующих отверстий добавить примитив отверстия другого диаметра. Сразу хочу предостеречь: наложение одного примитива на другой является не лучшей идеей.

Первая модификация модели заключается в удлинении Пластины 1. Это невозможно сделать без переопределения базового примитива. Чтобы избежать путаницы, большинство пользователей в этой ситуации просто добавит еще одно тело, имеющее форму прямоугольного параллелепипеда, как это показано на рис. 4. Теперь дерево построений содержит четыре шага, а параллелепипед для удлинения пластины является четвертым и последним примитивом в структуре. И снова не следует забывать о порядке построений.

Модификация №2: перемещение отверстия

Следующее изменение – перемещение находящегося на переднем плане отверстия при соблюдении условия, чтобы оно располагалось на таком же расстоянии от торца, как и второе отверстие (рис. 4). Как уже отмечалось выше, до тех пор, пока оба отверстия определены с помощью одного и того же примитива, мы не можем двигать одно отверстие, не двигая при этом и другое. Чтобы произвести желаемую модификацию, пользователю необходимо вернуться к дереву построений и внести изменения в примитив отверстия.

Результат попытки переместить существующий примитив (то есть, оба отверстия) показан на рис. 4. Если вы обладаете опытом использования Hybrid-моделлера, то наверняка поймете, что случилось. Примитив №4 (тело в форме прямоугольного параллелепипеда для удлинения пластины) в структуре дерева построений появляется позже, и располагается он ниже примитива отверстия. Поэтому, когда модель будет регенерироваться после перемещения отверстия, примитив №4 частично перекроет собой примитив №3. Чтобы решить эту проблему, надо изменить порядок КЭ и выполняемых операций в

дереве построений таким образом, чтобы примитивы пластины и удлиняющего его прямоугольного параллелепипеда разместились до примитива отверстия. Прежде чем начать, необходимо разобраться, какие дочерние элементы при этом будут затронуты. В нашем примере таких нет, но в реальных проектах всё будет не так просто.

Приведенный пример может быть использован в качестве доказательства того, что *Hybrid*-моделирование является неэффективной и непригодной технологией. В действительности же просто нужен соответствующий уровень подготовки пользователя. Надо отдавать себе отчет, что в работе с такими системами требуются опыт и мастерство. Это сродни освоению языка программирования. Как только вы овладели им, то можете сделать многое.

Разработчики систем *SolidWorks*, *Solid Edge* и других предприняли значительные усилия, чтобы снабдить пользователя инструментарием и облегчить работу с элементами, которые участвуют во взаимоотношениях типа “родитель – потомок”. Процесс переупорядочения дерева построений может оказаться достаточно простым, если этому уделять должное внимание. Поставщики *MCAD*-систем ведут разработку интересных технологий автоматического ремонта дерева построений в случае нарушения структуры. Таким образом, пользователь сможет свети к минимуму связанные с этим проблемы.

Работа с моделлером без истории построений

В *CAD*-системах без дерева построений пользователь взаимодействует с геометрией напрямую. Сюрпризов при этом возникает не много. Впрочем, утверждение, что не все модификации проходят, тоже достоверно. Несмотря на возникновение некоторых проблем, в этом случае всегда действует принцип *WYSIWYG* – “то, что вы видите, есть то, что вы имеете” (*What You See Is What You Get*). Проблемы могут появиться в процессе трансформации сложных моделей с поверхностями произвольной формы, сопряжениями и коническими поверхностями. Обычно модификация подразумевает добавление и/или удаление поверхностей, а также изменение их местоположения с последующей подгонкой смежных поверхностей (растяжение или сжатие). В любом случае приходится работать с граничными условиями в рамках *B-Rep*-модели. По существу, попытка трансформации может стать причиной появления некорректных граничных условий и, как следствие, возникновения ошибок. Системы без истории построений теперь стали более интеллектуальными в плане обработки ошибок и информирования пользователя о возникших проблемах.

В заключительной части статьи, мы разберемся с тем, что означает всё вышесказанное применительно к реальному процессу разработки изделий. ☹

(Окончание следует)

◆ Новости ◆ События ◆ Комментарии ◆

Поглощения на рынке САМ-систем

✓ В начале ноября 2007 года **Planit Holdings** (www.planitholdingsplc.com), в состав которого входят компании, специализирующиеся на разработке программного обеспечения для проектирования и производства, сообщил о приобретении английской компании **Pathtrace plc** (www.pathtrace.com), известной своей САМ-системой *EdgeCAM*. В результате этого поглощения группа *Planit* становится крупным поставщиком систем подготовки производства; её доход составляет порядка 51 млн. долларов, а размер клиентской базы – порядка 50 тысяч.

К брендам группы *Planit* относятся продукты *AlphaCAM*, *Cabinet Vision*, *CabinetWare*, *Fusion*, *Radan*, для машиностроения и деревообработки, а также *Planit Enterprise* – система управления бизнес-процессами, и другие разработки.

EdgeCAM – основной продукт компании *Pathtrace* – автономная система создания управляющих программ для станков с ЧПУ, которая дополнит имеющиеся у *Planit* программные продукты для машиностроительной отрасли. Наличие у компании *Pathtrace* офисов в Англии, Китае, США и Японии должно облегчить продвижение в этих странах всех САМ-систем группы *Planit*.

“В целом, бизнес *Pathtrace* хорошо сочетается с бизнесом нашей группы. Мы ожидаем увеличения нашей доли на рынке систем проектирования процессов обработки деталей, общий объем которого оценивается в 1.2 млрд. долларов”, – отметил **Bryan Pryce**, главный операционный директор *Planit*.

✓ В начале января 2008 года израильская компания **Cimatron Ltd.** (www.cimatron.com), поставщик интегрированных *CAD/CAM*-решений для инструментальной и обрабатывающей промышленности, сообщила о приобретении американской компании **Gibbs System, Inc.** (более известной как *Gibbs & Associates*), разработчика САМ-системы *GibbsCAM*.

В США образована дочерняя компания *Cimatron*, в которую и волеется *Gibbs System*. Президентом и *CEO* новой компании станет **Bill Gibbs**, основатель, владелец и руководитель *Gibbs & Associates*.

Объединение в одном портфеле *CAD/CAM*-системы *CimatronE* и решения *GibbsCAM* (www.Gibbscam.com) позволит новой компании расширить спектр продуктов, предлагаемых для обрабатывающей промышленности, и закрыть все сегменты рынка САМ-систем, включая разработку пресс-форм и штампов, подготовку управляющих программ для 2½-5-осевой механообработки и для многоцелевых токарных станков.

По предварительным подсчетам, общий доход *Gibbs* в 2007 году составляет порядка 12 млн. долл. Компания работает на рынке более 20 лет, её девиз: “Мощная простота. Простая мощь”.

По мнению **Rimon Ben Shaoul**, руководителя *Cimatron*, каналы распространения обеих компаний хорошо дополняют друг друга. Предполагается, что годовой доход объединенной компании достигнет 40 млн. долл., что позволит *Cimatron* укрепить свое положение на рынке и войти в *Top10* ведущих поставщиков САМ-систем. ☹