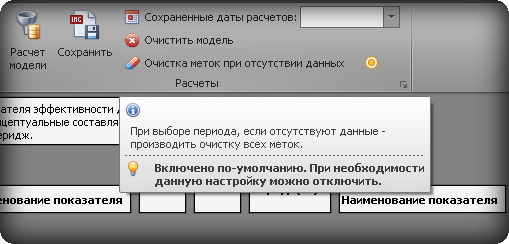
1. **Добавлен контрол checkbox «Очистка меток при отсутствии данных».** К нему добавлена всплывающая подсказка однозначно определяющая назначение установки флага. Выглядит это вот так:



При открытии окна факторного анализа по – умолчанию этот флаг уже установлен. При наличии этого флага, как уже ранее указывалось в файле «примечания от 16.06.16», будет производиться очистка данных всех меток в следующих случаях:

* При отсутствии данных по выбранному году;
* При очистке всех данных через кнопку «очистить модель»;
* При удалении какой-либо даты из сохраненных дат расчетов;
* При возникновении программных ошибок по какой-либо из причин, все Exception перехватываются в блоках try {} catch {};
* При расчете модели, в случае отсутствия данных или возникновения ошибок при расчете.

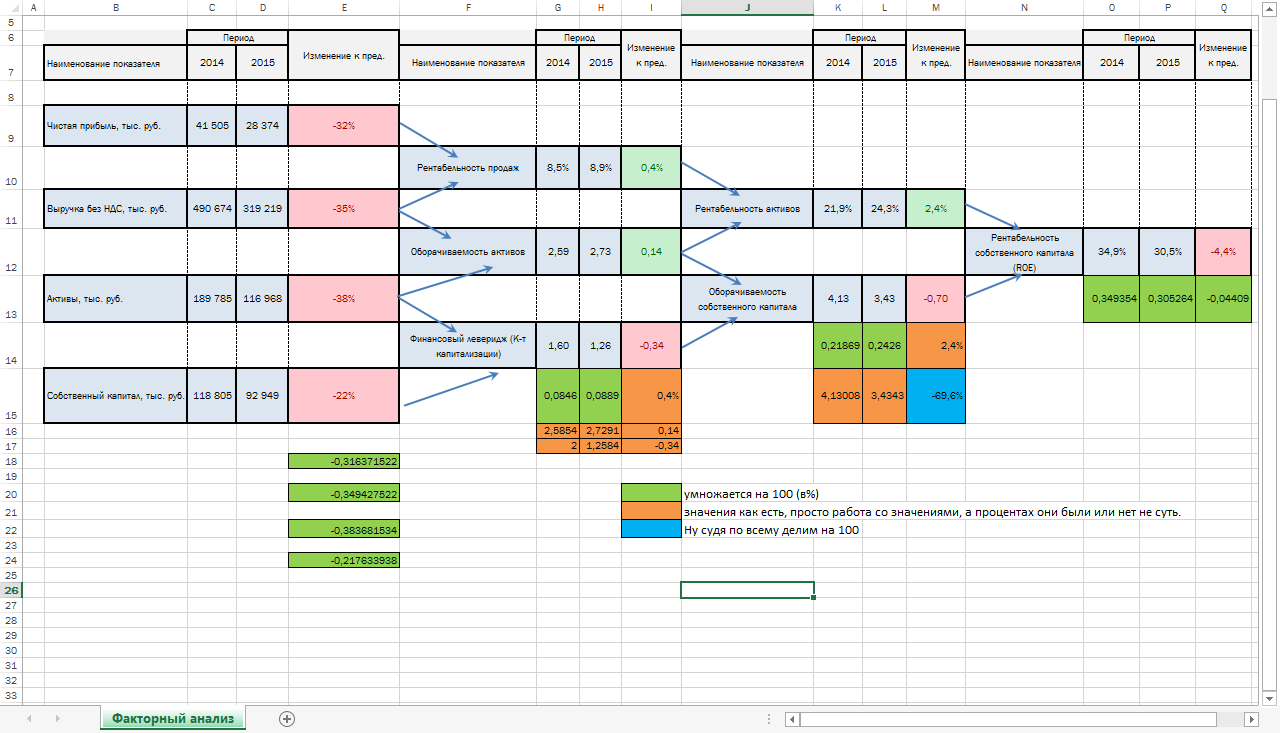
Данный подход позволяет наилучшим образом учитывать возможные ошибки, как и при действиях пользователя, так и программные. По большей части – это защита пользователя от самого себя. Если следовать линейному механизму работы: выборка начального, конечного периодов, расчет модели, то ошибок нет, но если начинать тыкать во все подряд, то учитывать все возможные комбинации ходов пользователя для корректной работы всех элементов достаточно трудоемкая задача.

Неудобство, которое мне самому не нравится, возникало в следующем случае. Если в одном из периодов год выбран корректно и данные уже загрузились в ракету, но при этом в другом периоде по указанной дате нет данных (или ошибка, что маловероятно), то запускается очистка всех меток. Т.е., включая те, что содержат корректные данные. Поэтому, приходится выбирать повторно. Это, разумеется, время и дополнительные клики мышью, хотя и избавляет наверняка от всех проблем.

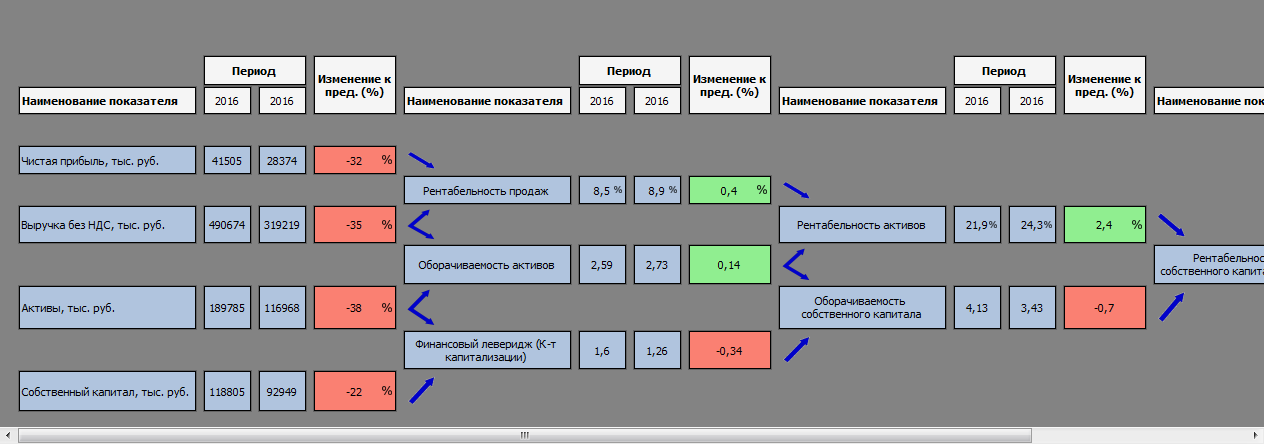
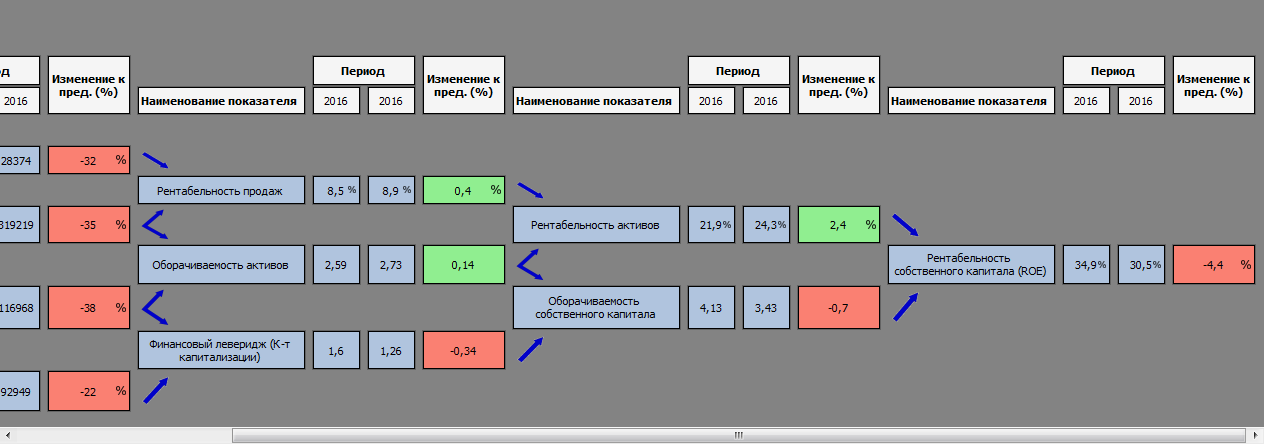
Поэтому сейчас я добавил контрол с выбором флага. Если его снять, то очистка производиться не будет. Подправив все обработчики, добился стабильной работы по всем возможным сценариям работы. И самое главное, при отсутствии данных, если были корректные показатели в другом периоде, ничего не сотрется. Кнопка очистки стабильно работает в независимости от того какой флаг выбран. Тем самым, каждый пользователь сам определяет, как ему удобней работать. Так как первый вариант в теории (при каждой непонятной ситуации делать очистку) мне кажется более стабильным, то и поставил наличие этого флага по – умолчанию.

Взаимосвязанные проверки всегда идут при выборке любого из периодов, при расчете модели, при удалении данных.

При выборе периода проверяется наличие данных в базе, если данных много, то вызов формы для выбора группы показателей. При расчете модели, условно по блоках, проверяются все части ракеты (4 расчетных блока – 4 метода) слева направо, где в каждом методе свои проверки на наличие данных, на корректность данных, так как, в самую большую часть у нас сначала идет загрузка из базы, а потом чем ближе к вершине, тем больше расчетов, которые опираются на результаты расчетов в предыдущих метках.

1. **Кнопка очистить модель**. Тут единственное дополнение, что кнопка одинаково очищает все метки, в независимости от флага и так же я добился очистки еще и выбранных значений периодов. Ранее, при очистке, автоматом срабатывало событие EditValueCheched, которое запускало свою череду проверок и загрузку данных, что вызывало ошибки. Сейчас, я дополнил сами методы изменения значения в этих двух выпадашках и очищает полностью все.
2. **По расчету самих меток**. В чем было несоответствие понял. В екселе автоматически учитывается расчет ячеек по их типу, а я просто задал формулы и там, где расчет в процентах умножал на 100, согласно полученным значения из продублированных формул под ракетой. На скриншоте показывал ранее. Вот оно самое:

А когда решил проверить расчет по екселевским тестовым значениям, то увидел, что и где не так. В итоге, я привел расчеты в полное соответствие с екселем, также обратил внимание, что я по- умолчанию, округлял до сотых все значения, что дает менее точные значения по мере прохождения по ракете. Ведь если для расчетов брать округленные значения, то есть небольшой процент потерь точности. Чтобы избежать этого, во всех расчетах я убрал округление и вынес для этого отдельную функцию, которая округляет все показатели после завершения расчетов. В этой же функции также задал до какого разряда округлять каждое значение. В екселе, обратил внимание, что где-то идет округление до целого, где-то до десятков, где-то до сотых. Привел в полное соответствие с екселем. Результаты на скриншотах:



1. **Следующее пояснение – работа с формой выбора группы показателей**. Ниже скриншоты:

* При открытии формы;
* При выборе показателей;
* При закрытии.

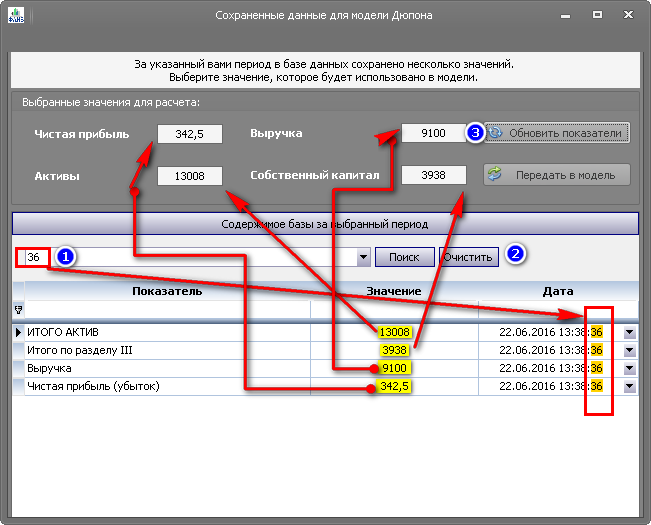
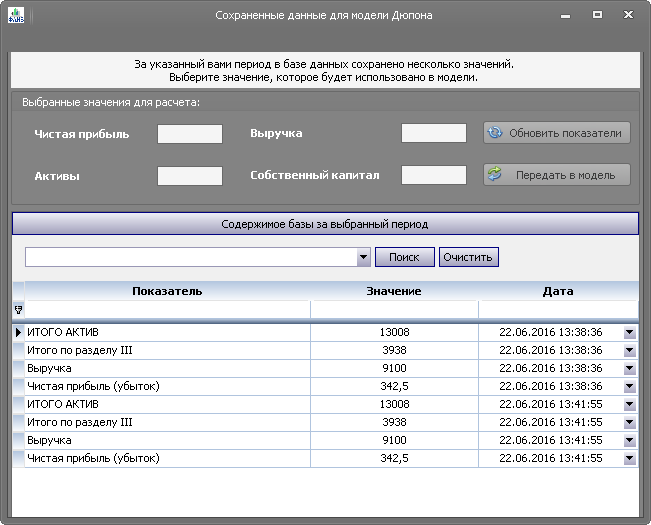
При открытии формы убрал заполнение текстовых полей текущими значениями из базы. Также я закрыл их для редактирования, обработал перерисовку, т.е., что-то случайно внести никак не получится. Какая основная концепция. Мы заходим, фильтруем по дате, получаем нужные нам 4 показателя, ни 8, ни 12, а именно 4 и нажимаем обновить, тем самым записываем 4 показателя в метки и при нажатии на передать закрываем форму, при этом вставляя в ракету нужные числа. А дальше уже идет расчет модели как обычно.

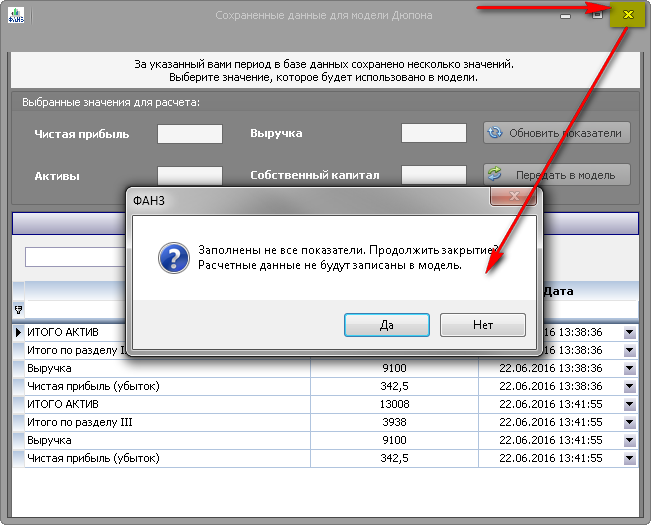
Т.е., я жестко задаю положение в таблице нужного мне значения. И именно поэтому нужно фильтровать данные, т.к., расчет идет по первым 4-м строчках. Когда за год будет туча значений, то другого способа выбора данных я не могу придумать. Это самый простой и быстрый. Работа с фильтрами и все. Вся суть формы.

Как видно на скриншоте, я добавил для столбца с датой вывод минут и секунд (тоже самое сделал в динамическом отчете). Чтобы удобней было фильтровать. На этом моменте дало слабину самое первое видео по работе, когда у меня были данные за 2 расчета, по одной дате. Вывел строку поиска, которая по – умолчанию скрыта (тоже самое и в динамическом отчете).

При закрытии, на случай если пользователь ошибся или передумал дал возможность дать обратный ход. Изначально я продублировал работу кнопки «передать в модель».

Полагаю, логику работы еще более упростил и максимально сократил варианты совершения ошибок. По крайней мере, из механизмов типовой работы, постарался все предусмотреть. Если что-то появится еще, то просто сделать очистку данных или переоткрыть форму и повторить свои шаги для расчета.

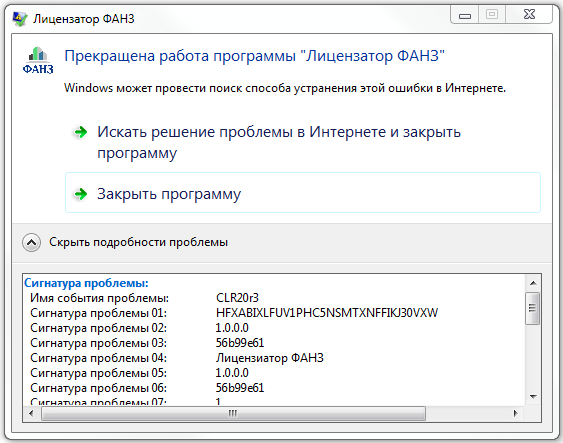




1. **Знаки процентов в определенных ячейках.** Как уже видно на скриншотах, во всех ячейках, где в екселе указаны проценты в программе я их тоже вывел. Я сделал в фотошопе макет процента с разными размерами, добавил в качестве изображения метки с ориентацией расположения справа. Такое колдунство обусловлено тем, что если ексель варьирует типами ячеек и их выводом, то тут это сделать в разы сложнее и чаще в целом неэффективно. В программе я работаю с числами, я их получаю, конвертирую, считаю, снова конвертирую, то в текст, то в числа, то снова в текст, а потом снова в числа. Поэтому лишние знаки для каких-то определенных ячеек – лишняя обработка. Вышел из положения свойствами самой метки. Изображение на текстовое содержимое не влияет.
2. **Ошибка при закрытии приложения.**

В примечаниях на 30.05.16 в последнем пункте уже упоминал об этой ошибке. Текст самой ошибки имеет следующую неизменную структуру:

*Необработанное исключение по адресу 0x731A4D66 в ФАНЗ.exe: 0xC000041D: Во время обратного вызова пользователя обнаружено необработанное исключение.*

Данная ошибка может проявляться как в основном приложении, так и в дополнительных: лицензиаторе или деинсталлятора. Как уже писал, у меня эта ошибка стала постоянно валиться при компиляции проекта. Долго искал причину, нашел виновника в виде конфликтов, используемых dll разных разрядностей. Те методы решения, которые можно было применить- применил. При компиляции валиться полностью перестало, уже больше месяца стабильно работает. При закрытии программы воспроизводится реже, но полностью проблема не ушла. Даже на последнем видео один раз поймал ее. Я тут уже ничего не сделаю. Это конфликт использования dll под x86 в x64-ом приложении в x64-ой системе, которое должно адекватно и полностью работать и под той, и под другой разрядностью. Какая именно из dll шалит узнать проблематично, с кем вступает в конфликт, что при закрытии выпадает тоже проблематично. Поэтому если такая ошибка и будет иногда проявляться, то просто закрываем и все, никак на данные в базе, на саму программу или ее механизмы работы это не влияет. Выглядит это окошко всегда одинаково:

Имя события **CLR20r3** – говорит о том, что ошибку порождает net. framework. Это однозначно. Версия значения не имеет, хоть уже устаревающая 4.0, хоть 4.6.2.

Т.к., все источники утверждают, что это конфликт разрядности, то могу предположить, что полный переход всего приложения и сопутствующих сборок на x64 – версию с использованием в x64 – ой системе может решить проблему на корню. Кстати, быстродействие приложение это точно повысит. Но это такой топорный метод. Хотя могу и ошибаться, я понятий не имею под какую разрядность заточены и насколько комфортно и корректно чувствуют себя в разных системах компоненты devexpress, но **CLR20r3** отношения к devexpressне имеет. Но пока часть сотрудников и потенциальных пользователей работают под x86, это невыгодный шаг. Так просто закрыть окно и работать дальше, а так можно потерять часть пользователей. Я вообще не удивлюсь, если люди еще на windows xp сидят и не жужжат.

В будущем я планирую сделать диагностическую утилиту, которая будет за указанный промежуток времени имитировать запуск фанз, закрытие, снова запуск и закрытие и так по 1000 запусков к примеру, при отлове ошибке – запись времени в лог, и параллельно в момент тестирования будет работать утилита для сбора лога по используемых компонентами в системе, т.е., все программа и все dll, откуда запускаются, когда запускаются, кем и в какое время. По итогу, можно будет попробовать понять на стыке времени падения, что происходило и выцепить-таки проблемные dll. И то если в этом будет такая необходимость. X86 уже морально устарела и абсолютно все делается под x64.

1. **Крякалка для пароля администратора ФАНЗ.** Я, когда проверял под админом программу снова забыл какой пароль когда-то сохранял и решил сделать проще. Написал крохотную программу, которая делает кодирование введенного текста по тому же алгоритму, который используется в ФАНЗ. Т.е., если забыли пароль, то можно сгенерировать что угодно и результат кодирования скопировать в файлик pas.txt, который лежит в корне программы. Сейчас административные модуль не столь актуален, но возможно при продажах он расширится или даже преобразуется, не исключено, что существующий функционал может перебраться в режим работы админа, а у простых пользователей будет частично ограничен.