ADAMS/Car-AT

in The Chassis Development

at BMW

Ewald Fischer, BMW, 14. November 2001

Introduction

BMW's chassis department has been using MDI software for about 15 years now, and was (and still is) working on the development of ADAMS/Car. This paper describes the achievements and the deficits in the area of 'virtual prototyping' in three sections:

- Requirements
- Current state-of-the-art
- Current developments and plans

Requirements

BMW's products are cars, and so everybody working at BMW has from time to time to answer the question what - or how much - he/she contributes to the development process, which eventually leads to cars driving on the road.

Now, a group or person working on 'virtual prototyping' should not have any problem at all in justifying their work - provided the term 'virtual prototyping' keeps its promise. But one has to say right here and now that not all of these promises can be kept. Yet since virtual prototyping has in fact shown some growth over the years, it obviously can make some valuable contributions to the development process. It makes these contributions to the second of the three threads of the development process

- Geometric Integration
- Functional Integration
- Production Integration

by generating statements on how well the actual design of the product fits the requirements in the following areas:

- Kinematic
- Elasto-kinematic
- Comfort
- Dynamic Loads
- Special Topics

The following quite simplified picture shall illustrate a few statements on virtual prototyping:

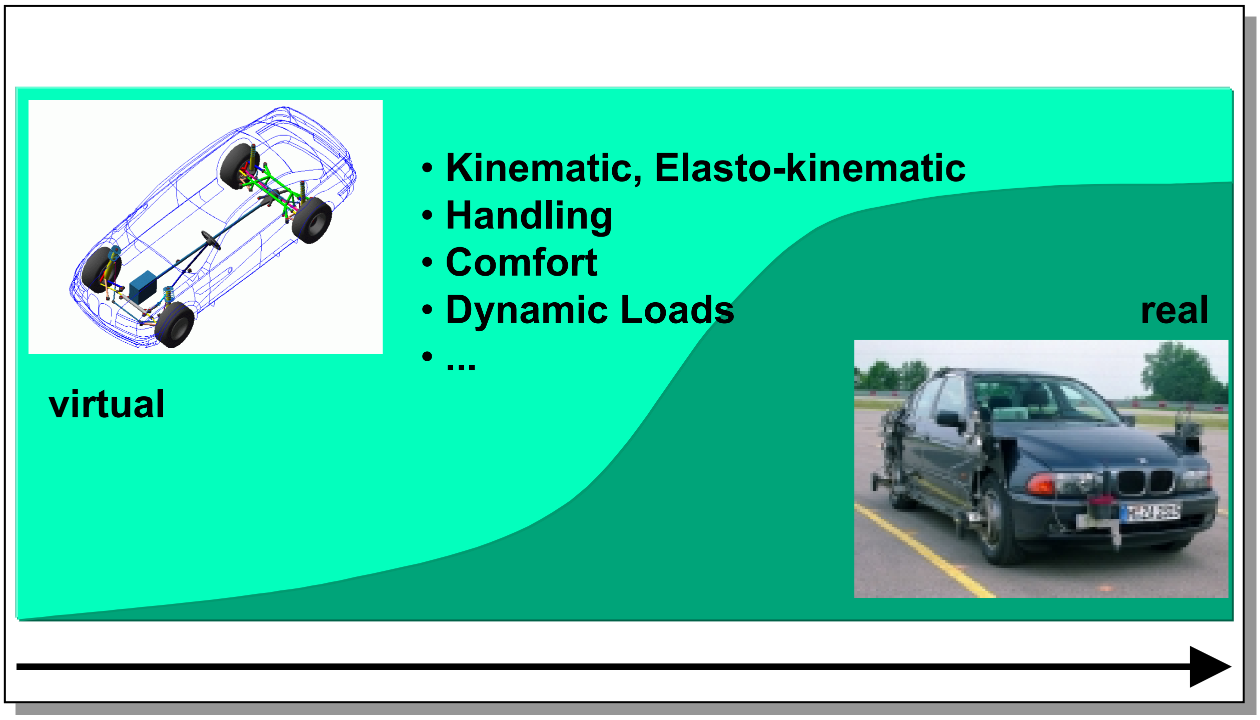

Figure "Functional Integration"

- Today 'Virtual prototyping' is part of the process,
- We have started to push the 'hardware prototyping' wave further back in time,
- There will always be hardware prototyping.

The most important process partners for the members of the 'virtual prototyping' group are quite naturally departments doing design and test. In the development teams the virtual prototyping engineers do not only play the role of a service supplier, providing simulation results if needed. They also play a role as a tool-supplier, encouraging colleagues from the design and the test departments to use the tool themselves. Obviously supporting such casual users - i.e. non-experts also known as 'end-users' - requires special tool capabilities as will be explained later.

Additionally several interfaces have to be supported. First an interface to the big unified company-wide database, also known as PDM system, second interfaces to CAD systems and third interface to CAE-Tools other than ADAMS. The current state of these interfaces will also be explained below.

<u>State of the Art</u>

History

BMW has started to use MDI Software in the middle of the 80's, and it began with the ADAMS/DMP software. For those who do not know DMP, i.e. for the younger colleagues, one must state that DMP never was a tool you long for. Yet if the only alternative to DMP was to write large ADAMS decks by hand, you better take what you can get. On the other hand, if DMP would have been nothing but an old and ugly ADAMS deck generator, it would not have survived so long, until the end of the 90s. So what were the real features of DMP? In my opinion one can condense them to three points:

- Simple language & concept
- Re-usable modules,
- Separation of topology and data.

In none of these areas did DMP offer high-tech-solutions, only low-tech, yet it took a really long time to replace DMP with other tools.

The first attempt to develop a successor for DMP was started in 1994 with the development of ADAMS/Car, based on the old ADAMS/View GUI. Since the result was not satisfactory - people still used DMP to get their work done - BMW launched a second attempt in 1997 under the name ADAMS/Car-AT (where AT stands for advanced technology). The project setup was not 'yet another customization project', but to jointly develop new functionality according to the customer's needs and to allow MDI to market this later.

The goal was to make ADAMS/Car-AT a widely used tool with

- large functionality,
- good usability,
- appropriate standardization
- high reliability,
- high performance,
- customizability.

One can argue about the degree to which these objectives have been achieved. But it is a fact that today ADAMS/Car-AT is BMW's strategic MBS tool for chassis development.

Asking for the most significant change between 15 years ago and today, one could list the many new features developed here and there - flex bodies, SI2-integrators, GSE, GUI Tool kit, controls, PPT, or whatever - but one can also draw the following picture

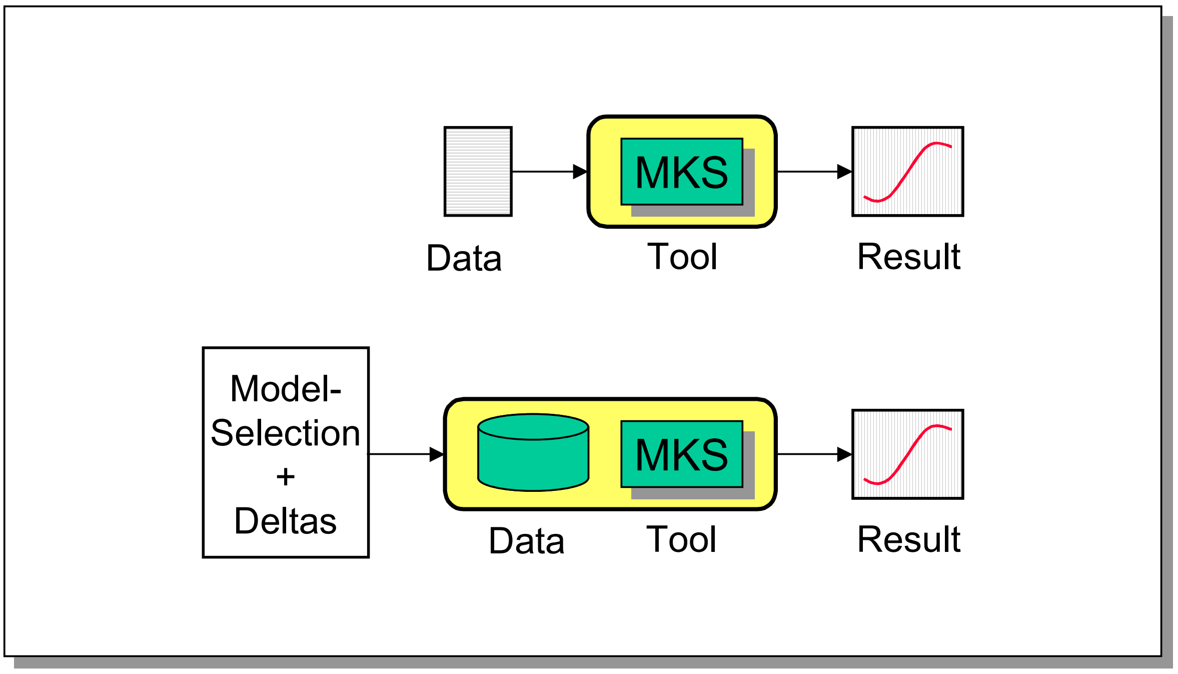

Figure "old/new"

Today our users expect that they can re-use almost everything: models, data, virtual experiments. What they do is select the right model, make the right changes and run the right experiment - ideally after having found the right documentation. And the news that you can work this way has spread around. I have seen examples of BMW test engineers, who were famous for their technique of putting washers here and there in the (hardware) cars and then driving around, now sitting in front of a computer screen. So virtual prototyping has really started.

But be aware that 'the tool' I am talking about is no longer just an MDI product. It is a product consisting of a external software component, known as MDI software, and an internal component, known as data and models.

Interfaces

As mentioned above a tool needs to have interfaces to other tools in order to be useful in the development process - at least as long there is not a mother of all tools covering the whole process.

Currently the hottest interface topic is the PDM-interface. The goal is to have a single huge database for everybody in the company which can answer almost any question. But not only questions posed by human beings: CAE tools would ask questions on mass, inertia, stiffness, damping etc., and CAE tools would store answers in the PDM system such as suspension or vehicle dynamic characteristics. We are currently working on this and we have the very first pieces in place, yet there is still a long way to go.

And as long as we have not reached the end of this way, we have to work with the "poor man's PDM system", i.e. the /Car-DBs. /Car-DBs are no leading-edge technology in database business, and it takes a lot of effort and additional tools to provide a minimal set of database-like facilities such as data history or version control. But they are an absolutely necessary component of the tool and a prerequisite of its success.

A non-success-story, at least until today and at least at BMW, is the topic of CAD interfaces, namely CATADAMS. Although this tool was advertised and although it was supported in /Car-AT, it was and is hardly used. Obviously the supply did not meet the demand. One could guess that the main reason for this is the constraint that users have to master two worlds, CATIA and ADAMS/Car. This is definitely true, but it may not be the only reasons. It is also true that virtual prototyping sometimes has to give statements on the concept or design of a car, when this does not yet exist as CAD model.

The third group of interfaces, those to other CAE tools, shows more elements of a success story. The generation of FE load cases as NASTRAN decks from compliance analyses in ADAMS/Car-AT is a standard technique in the development process, reliably usable by any type of user. The usage of FE models as flexible bodies inside ADAMS simulations is also quite standard, although it is still in the domain of expert users.

Another important element in BMW's development process is the ability to generate input decks for a proprietary tool called "2Spur". 2Spur's functionality can well be compared to what was (and is?) known as "Conceptual Suspension" in ADAMS/Car: Based on an description of suspensions by curves - rather than parts and joints - this tool can be used to generate statements on the vehicles handling behavior. We see 2Spur more as a complement to ADAMS/Car-AT than a competitor: 2Spur has limited capabilities, but it is fast, simple and totally dedicated to the topic 'vehicle handling'. ADAMS/Car-AT is definitely not as fast, does serve much more different purposes and is still growing in functionality. Together with the ability to generate 2Spur input decks from hardware tests with the "K&C test rig" (Kinematics and Compliance) as shown in the following figure, the 2Spur tool offers a very efficient way to do virtual prototyping.

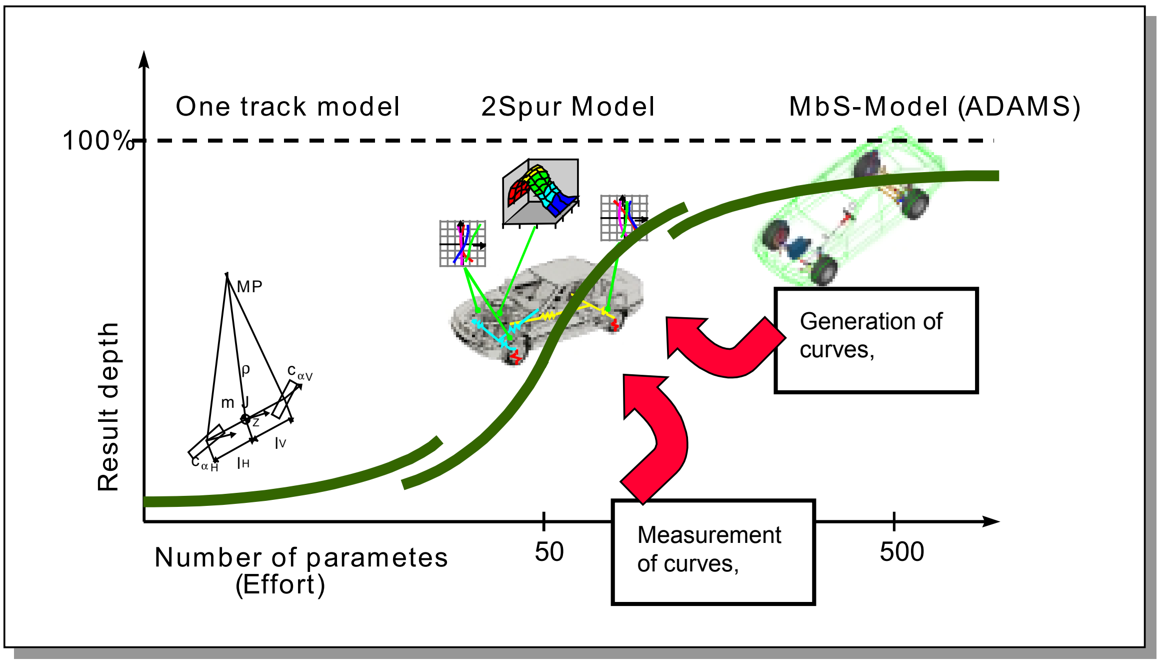

Figure 'Model Classes"

Standardization

If you want to have re-usability you have to pay the price of a certain modularization and a certain standardization. Since the old ADAMS/Car we have a model-hierarchy according to the following scheme:

Components: Bushes, Tires, ...
Subsystems: Suspensions, Steering, Wheel
Vehicles

So nobody who has ever heard about /Car will be surprised to see a figure like the following:

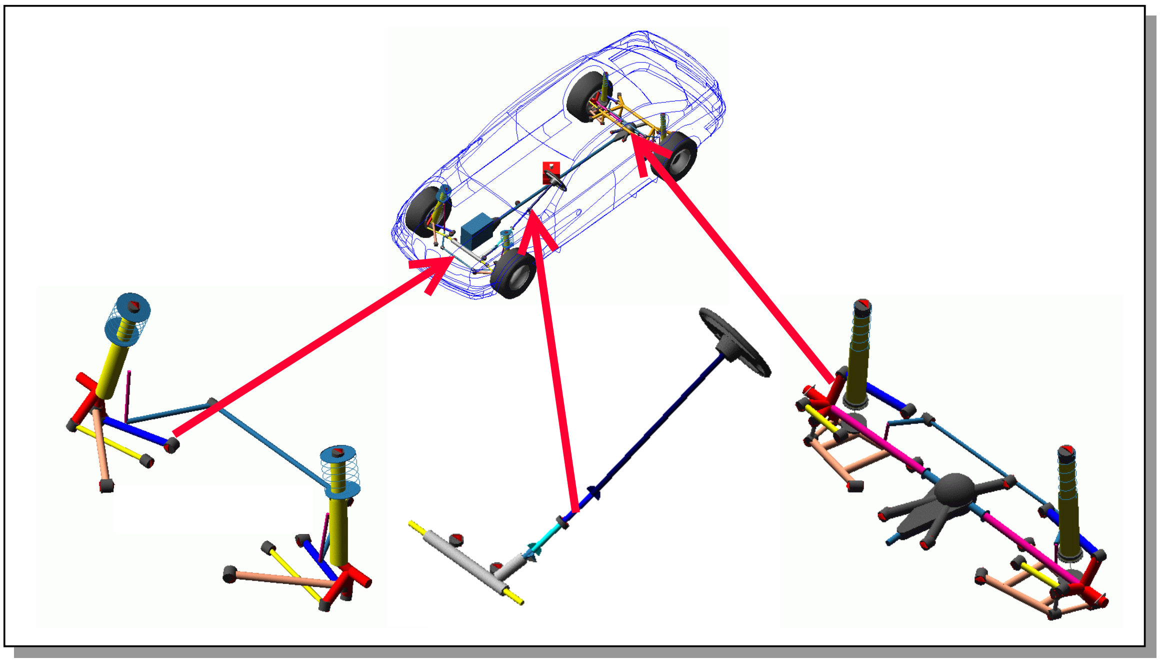
Fig "Model Hierarchy"

Now although this scheme is mainly dictated by ADAMS/View and not the result of a free design decision - which for example would have voted for a "roll bar" as a subsystem of a suspension - it is something we could live with - for the beginning. But its limitations hurt us more and more: Imagine for example that you have got or developed a nice little ADAMS model - just ADAMS! - of a bush with frequency dependent characteristics. Getting this into your ADAMS/Car model has now just become doable with the help of UDEs, yet making UDEs is a black art and requires lot of work.

Let me mention a vision at this point: If you have got this nice little bush model it shall not take more than making a declaration ("this is something equivalent to a BUSHING"), adding some hooks for the supply of data (not: programming macros) to make this usable in the ADAMS/Car context. The equivalent would apply to models of type "roll bar", of type suspension etc. In short: Everything is a submodel - even a bush. Comparing this vision with (my understanding of) MDI's development announcement gives the feeling, that this is not a totally crazy vision, but that one has to wait quite a few release cycles.

Another type of standardization goes back to the old /Car days. The distinction between 'models' and 'test rigs': Only models assembled with a test rig yield a ADAMS model which does something.

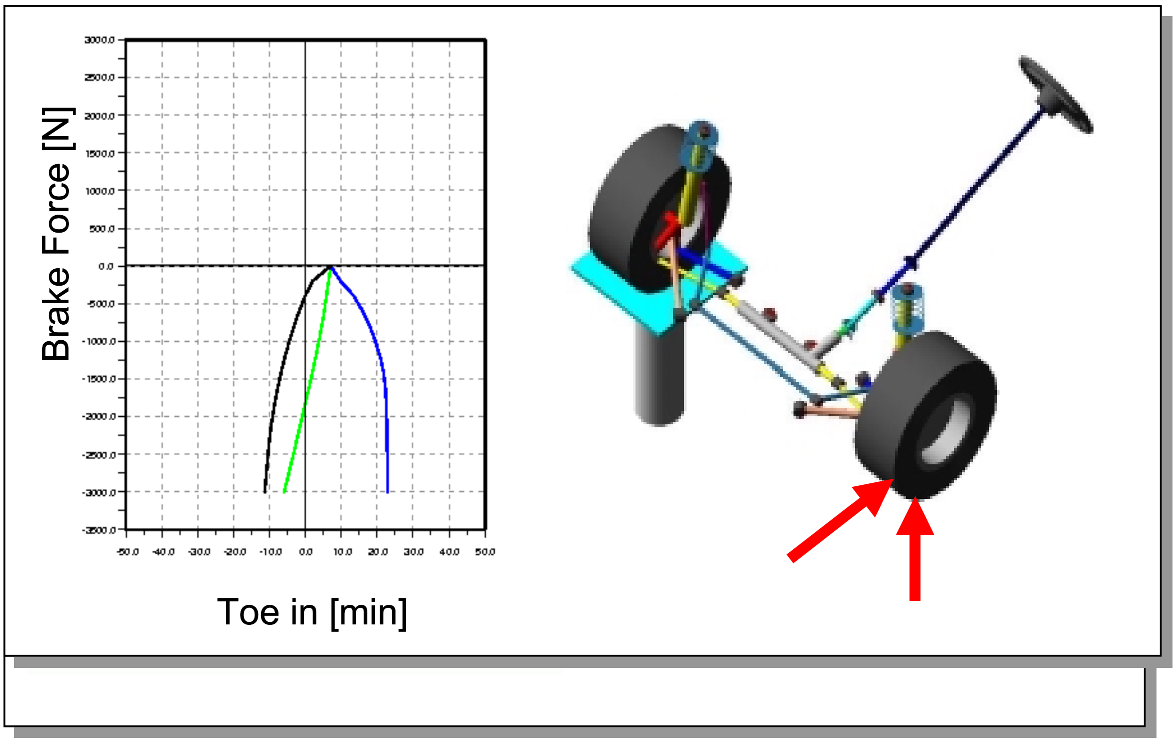

Figure "Virtual test rigs"

Although the term 'test rig' is a bit misleading in the context of full vehicle models, it did and does make sense. And - maybe the most important point - if you build your virtual test rigs (and their virtual test programs) with references to hardware test rigs in wording, graphics, etc. you have easier access to the world of designers and test engineers. That's also an indication for the reasons why we would rather have a few more different test rigs with dedicated simpler functionality - as in the workshops - than just a few complicated multi-feature test rigs. So we have a suspension test rig and a full vehicle test rig, test rigs for steering systems, 4-post-shaker, tires, bushes, etc. Ideally, we would have a test rig for any of the model classes we deal with ...

The next area of standardization follows logically from standardized models and standardized test rigs:

- Standardized virtual test programs
- Standardized result generation
- Standardized virtual test series.

The following figures just illustrate example for such standard applications:

Figure "Elastokinematik"

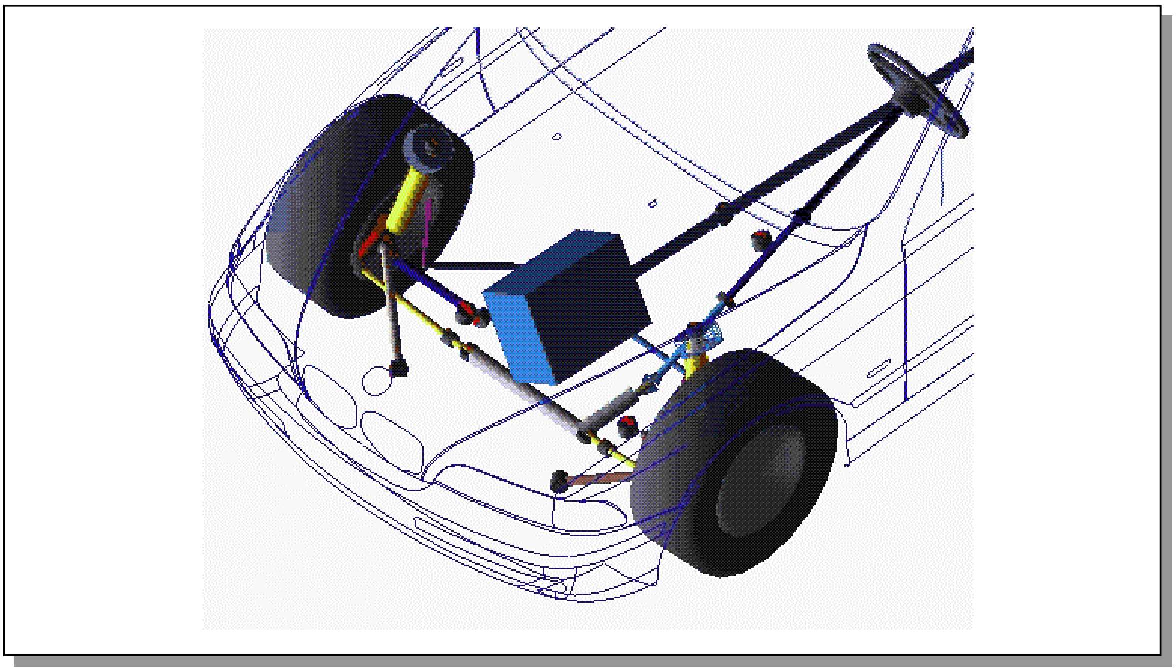

Figure "Schwingungsuntersuchung"

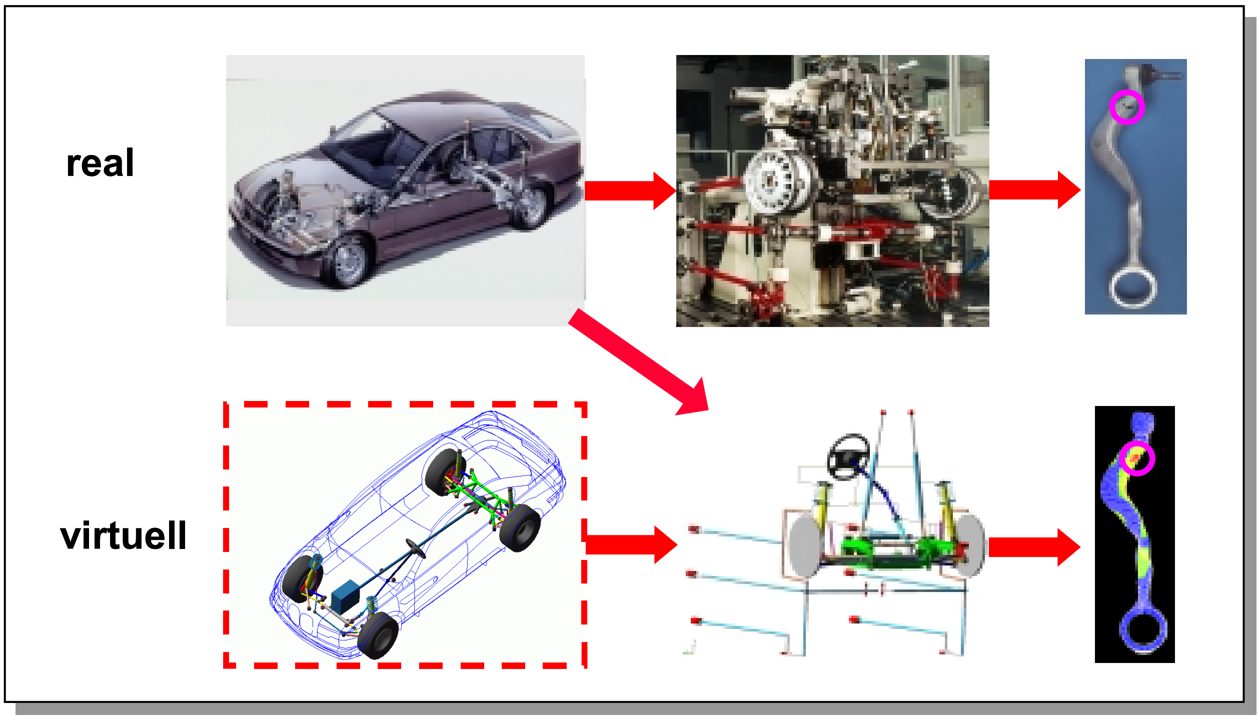

Figure "Lebensdauer"

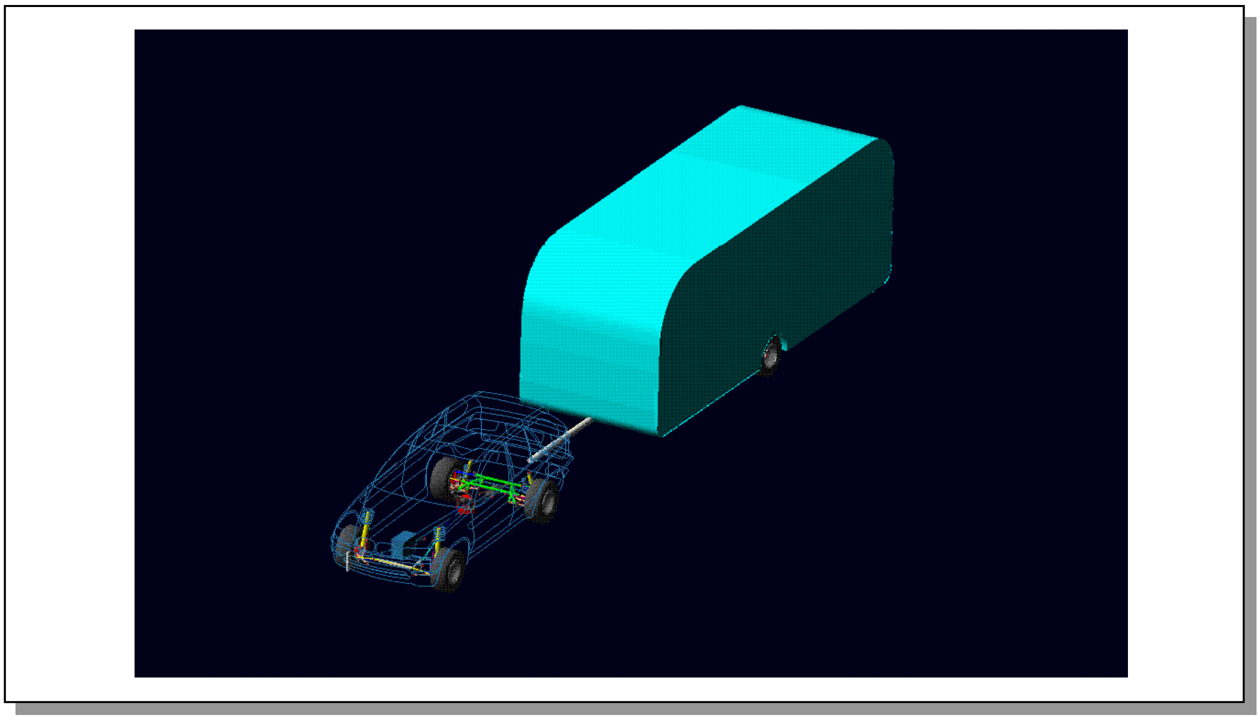

Figure "Pendelstaebilitaet"

Naturally standardization requires additional effort and makes you pay other penalties. First there is the effort in setting up rules, for example on how to build models and test rigs plus methods to check for compliance to these rules, the effort in developing test macros and standardized post processing methods covering as many needs as possible. Second you have the penalty to pay that there's a tendency to limit the possibilities of experts, who have to do a task, which is not covered by the standards, but who want to re-use the models or test rigs offered by the standard tool. To some degree this does not apply to /Car and /Car-AT, because it is still a fairly open system. But to a significant degree it is true, because only highly trained people find their way through these fences. Note that these people do not have to be trained with respect to MBS modeling or tires behavior or whatever, but trained in many languages (FORTRAN, C, View, C-shell, TEIM-ORBIT) and in the idiosyncrasies of the /Solver, /View, dcd and cdf files and alike. So there are not many of them.

Anyhow, here two examples were some non-standard techniques was combined with a standard Model

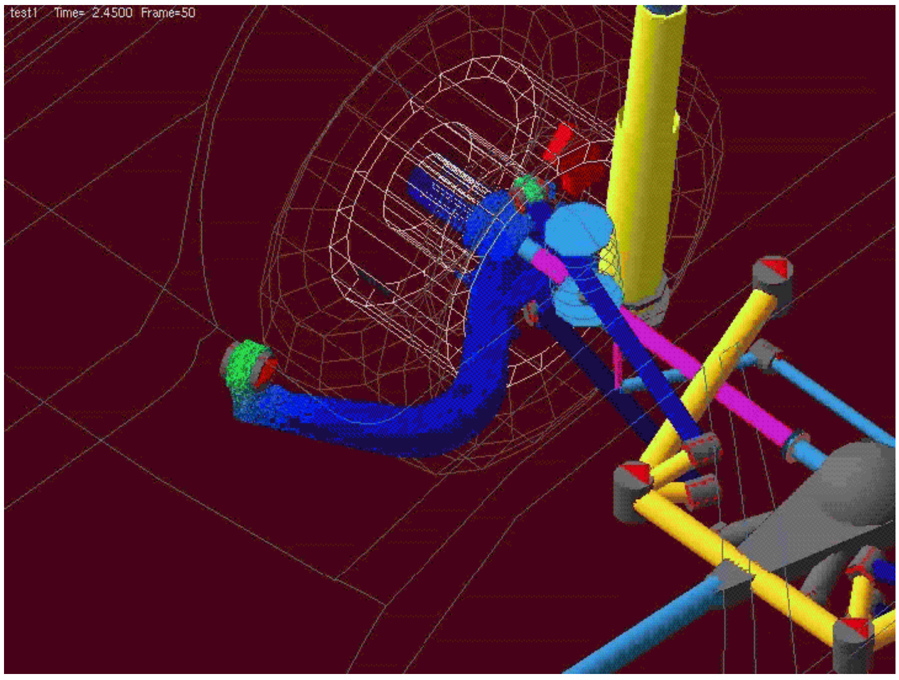

Figure "FlexBody"

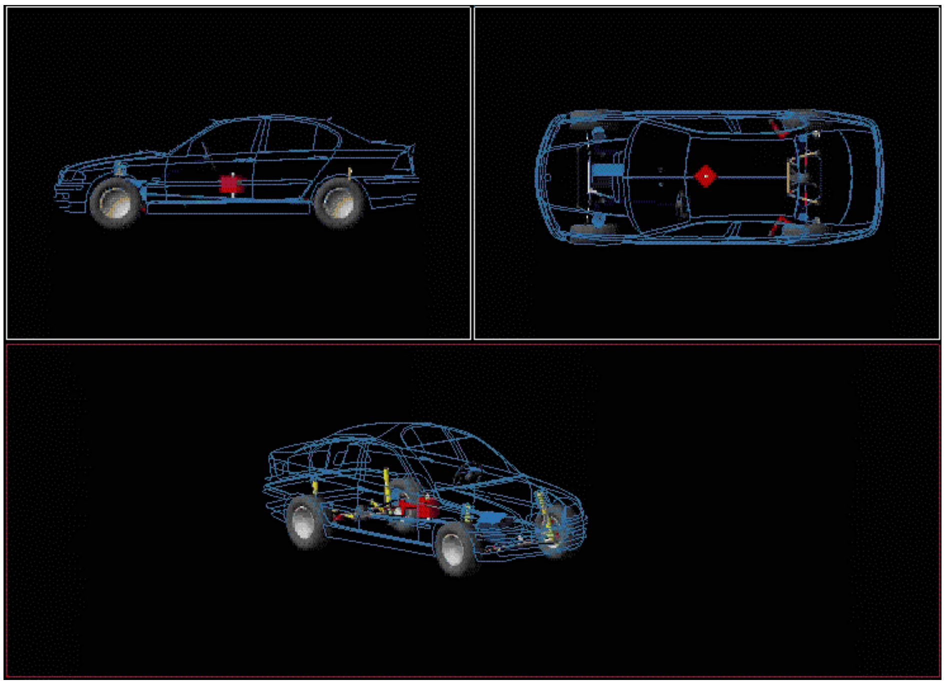

Figure "Special Tires"

Here is a question (not a complaint) to myself and to others: Is the increase in complexity justified by the increase in functionality? Or have we sometimes missed the very first rule I learned in my very first ADAMS training from Jim McConville? It was called KISS (Keep it simple, stupid).

Quality Control

All the efforts we have put into the standardization and re-usability as explained above would be useless if the users could not rely on the results they get. In other words: You cannot claim re-usability and ask users to take advantage of re-usable modules and methods, if you do not provide these in a high level of quality. Users who encounter crashes with a feature they have used successfully yesterday, or - even worse - if they notice different results now compared to results they have obtained with the same model two weeks ago, will not buy into that tool.

Sure, validation is an important aspect in this context and we do cover it, as the following picture illustrates. But validation is not sufficient for providing the quality required for a 'virtual prototyping tool'.

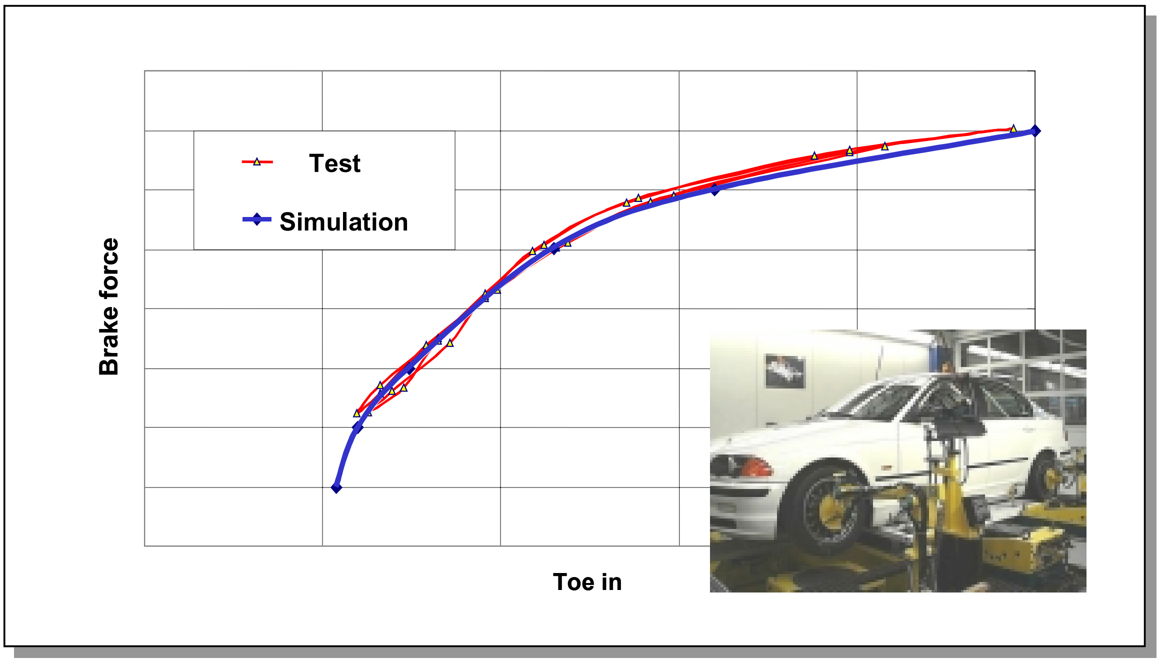

Figure “Validierung”

The consequences were two-fold. First we had to generate an awareness among all those users who “just” supplied models and data that they indeed “produce software” and that they take all the risks unfortunately associated with software production nowadays. Second we had to develop QA tools and methods to minimize these risks. Since then the regular application of these methods - not only when the software release number changes

- are an integral part of the maintenance process we have established at BMW for our virtual prototyping tool ADAMS/Car-AT. I must stress again that our word “tool” comprises more than just MDI-supplied functionality, but also our models, our data, our extensions and customizations. However, one cannot conceal that our regressions sometimes also catch problems which slipped through MDI's regressions.

The following figure shows this software process:

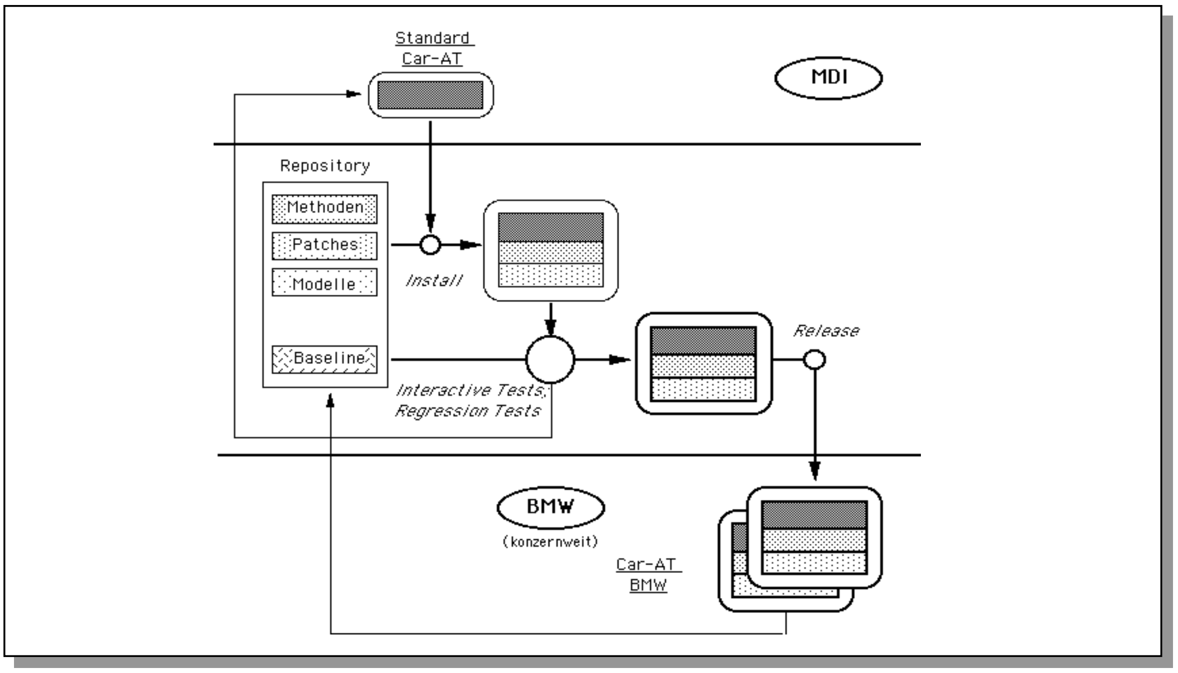

Figure "Software Process"

At this point I want to repeat my idea, that there is probably a demand for QA tools or QA service around ADAMS, which MDI could answer. (The final responsibility for the quality of the tool offered will nevertheless be with the customer).

Current Development and Plans for the Future

The interface to a PDM system is a hot topic at BMW as already mentioned, and the goal is, that data, which is not clearly 100%-tool-specific, has its ultimate storage location in the PDM system, and that both human beings and tools of any kind can access this data easily. For ADAMS/Car-AT this implies that some day down the road the /Car Databases are populated only by templates (and maybe things like UDE definitions), and that all other property files have vanished into the PDM system.

We have also made significant efforts to automate the execution of large test series in order to better exploit the potential of virtual prototyping: virtual cars or subsystem can exists in any number of copies, virtual test drivers as well, and they do not even get tired.

In this respect we are not only looking forward to the great unified tool resulting of the announced merger of ADAMS/Pre, ADAMS/Car and ADAMS/Car-AT, but also hope for some earlier improvements in MDI's software (for example in /Insight).

As explained above we still need more flexibility in modeling and easier modeling techniques in order to have truly scalable models with re-usable components: It must be possible for any user (or the tool itself) to replace all standard bushes in a suspension by a highly sophisticated bush model while re-using the existing topology and geometry. Note that writing UDEs is in our opinion not a technique which deserves the term modeling, but is a tough exercise in /View programming (although we do appreciate having this method at all).

We also need an easier, more reliable - in short end-user compliant - process for the handling of Flex Bodies. Additionally, we have to develop processes which allow us to incorporate control systems in ADAMS-models upon request. And we are working on a rating driver model.

As always we need better tires.

Now to finish let me mention a few basic deficits and conceptual problems for which we have not seen or heard a sufficient answer yet.

As already mentioned there is a need for more freedom in the model hierarchy. (And by the way, the same instance of a submodel shouldn't exist as two independent copies in memory.) What I understand is that with one level from top (submodels) and one level from the bottom (UDEs) the end of MDI's current technology is already reached.

We also believe that virtual prototyping for end-users implies that they can "measure" everything by point&click, after they have managed to select the right model and the right experiment. They should not be forced to ask experts or /View programmers to get a non-standard result. Combining such "measures" to yet another measure should not require more programming knowledge than the ability to write algebraic expressions (function expression). In this respect one can even observe a step back in /Car and /Car-AT compared to plain /View.

It was a smart-looking idea (not only smart-looking to MDI, but also to us) to use a 'lite' version of ADAMS/Driver as 'driving machine' for our standard maneuvers. However real life experience is that Driver-Lite is a heavy trouble-maker. The problem persists from version to version and does not increase the popularity of the tool nor the popularity of the software suppliers MDI and IPG.

End-user compliant applications like ADAMS/Car can be understood as dedicated layers built on-top of more general tools, such as ADAMS/View and ADAMS/Solver in our case. An interface between these different layers which consists of (1) an input path from /Car to /Solver and (2) a result path from /Solver to /Car is an incomplete interface. As Murphy is telling you, something is going wrong and therefore you have (3) to communicate the problem to the /Car user. But “communicate” cannot mean that you throw an ADAMS/Solver message file at him/her or store this messages silently in some log file. Communication means that you talk the language of the /Car user. I have been told for years that a "messaging facility" will provide the basic technology to cover this - and for years I hear that it should be there "real soon now".

Summary

The paper showed that MDI software is an integral part of BMW's chassis development process and that BMW takes advantage of the 'virtual prototyping' facilities provided. It also tries to make clear that there is still a significant gap between the potential of 'virtual prototyping' and its current use. It was explained that BMW has learned the lesson that running a virtual process is not done by buying from software suppliers only, yet it should have become clear also that it is often not easy to justify the amount of work we put into the tool rather than into our products.