KNOWLEDGE AND NEW PRODUCT DEVELOPMENT: PROBLEM SOLVING STRATEGY AND ORGANISATION

Markus C. Becker[a]
Francesco Zirpoli[b]

[a]Bureau d'Economie Théorique et Appliquée (BETA),
CNRS, France
becker@cournot.u-strasbg.fr

[b]Dipartimento di Ingegneria Meccanica,
Università di Salerno, Italy
fzirpoli@unisa.it

Session C-2

Abstract

Viewing new product development as a problem-solving activity, the focus in the article is on abduction as a problem-solving strategy. We identify the advantages of problem-solving by way of abduction, as compared to induction and deduction, present a rationale of how to decide between the different problem-solving strategies, and finally, draw on a case study of a main European car manufacturer to analyze how problem-solving by abduction is implemented in practice.

Keywords: New product development, problem-solving, abduction, simulation, experimentation, automotive industry.

KNOWLEDGE AND NEW PRODUCT DEVELOPMENT: PROBLEM SOLVING STRATEGY AND ORGANISATION

Markus C. Becker[a] and Francesco Zirpoli[b]

[a]Bureau d'Economie Théorique et Appliquée (BETA)
CNRS, France
becker@cournot.u-strasbg.fr

[b] Dipartimento di Ingegneria Meccanica,
Università di Salerno, Italy
fzirpoli@unisa.it

Abstract

Viewing new product development as a problem-solving activity, the focus in the article is on abduction as a problem-solving strategy. We identify the advantages of problem-solving by way of abduction, as compared to induction and deduction, present a rationale of how to decide between the different problem-solving strategies, and finally, draw on a case study of a main European car manufacturer to analyze how problem-solving by abduction is implemented in practice.

Keywords: New product development, problem-solving, abduction, simulation, experimentation, automotive industry.

Suggested track: Knowledge creation and innovation.

1 Introduction

New product development can be seen as a problem-solving activity (Iansiti and Clark, 1993; Iansiti, 1995; McDonough and Barczak, 1992; Thomke, 1998a, 1998b; Verganti 1997; Thomke and Fujimoto, 2000). Organising problem solving activity, however, is not at all trivial. The task is particularly challenging in the case of complex products (Nightingale 2000). A wide array of problem solving strategies is employed in new product development where many different components, inputs, and actors have to be integrated in the new product development process. Successful problem-solving typically rests on two factors: a strong scientific background, and experience in product and process applications. Verganti (1997) has used the term “systemic learning” to denote such learning that includes both sources.

In large firms, this distinction is often institutionalised in the form of R&D laboratories and component- or technology-based functions on the one hand, and project-based units on the other. These two bodies of knowledge tend to interact, yielding the benefits of both the application of laws (i.e. fluid dynamics, electronics, hydraulics, etc.) and the insights from experience (held in the form of “experiential knowledge”). This results in a loop of theory building and testing when a new product is designed, shown in figure 1. The implications of problem-solving by applying this loop are better and more innovative products (product quality), and reduced development cost (for more background on the relationship between science and innovation see Nightingale (1998)).

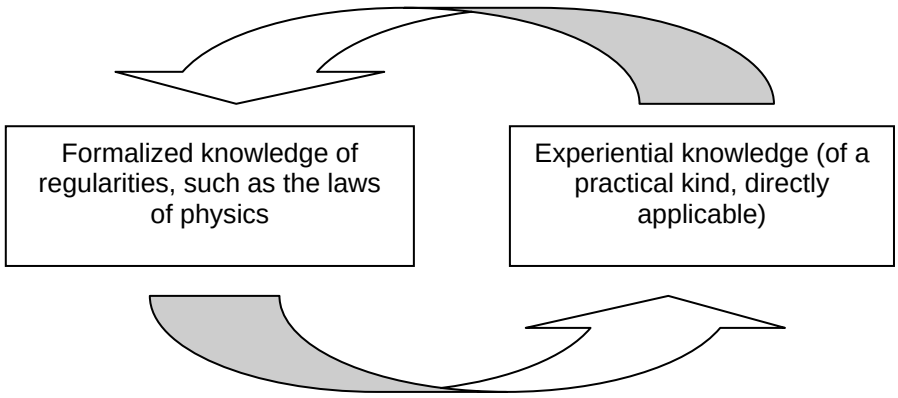

Fig. 1

A typical way of applying this loop is by making use of experimentation, i.e. the systematic testing of ideas. The trial-and error process triggered by experimentation greatly contributes to finding new solutions to the typical problems encountered by engineers during a development process.

As clearly mirrored by the literature, during the last fifteen years, however, the key to new product development success has been the ability of companies to shift the identification and solving of problems to very early phases of the product development process - a concept defined as front-loading (Thomke & Fujimoto, 2000). This has called for the development of experimentation processes based on the virtual simulation of reality, allowing the anticipation of many observations and engineering solutions in a reasonable time frame and at sustainable costs (Thomke, 2001a).

In this paper, we argue that adopting novel development technologies has an impact that goes beyond the mere improvement of product development performances. It

does, in fact, contribute to changing the nature of problem solving itself, thereby requiring a profound reconceptualization of engineering activities and their organisation. To support this argument, we first identify which problem-solving strategies are usually related to product development and show how *abduction*, as compared to *induction* and *deduction*, is particularly adequate for the engineering innovation process. We then present a rationale of how to decide between problem-solving strategies, taking into account the formalization and speed of change of the knowledge bases drawn on in solving the problems at hand. We finally, on the basis of a case study of a major European OEM in the car industry, analyze how problem-solving by abduction is implemented in practice and show that virtual experimentation is the tool most appropriate to sustain and foster *abductive* problem-solving. Finally, we draw conclusions for future research and for practitioners.

2 The basic problem-solving strategies applied in new product development

There are three basic strategies for solving problems: deduction, induction, and abduction. While the first two are well-known, abduction has received less attention in the literature on new product development. This is surprising, given Herbert Simon's observation that abduction 'is the main subject of the theory of problem-solving' (Magnani 2001, 94-5). In the remainder of this paper, we argue that abduction plays a central role in problem-solving in new product development, and that gaining a deeper understanding of abduction as a problem-solving technique can improve our understanding of how to best organise new product development activities.

Deduction can be interpreted as applying knowledge held in a general form, for instance of general laws (e.g. the laws of physics), to specific problems. These later are solved by understanding which general knowledge is available about specific problems, which laws govern them, and by applying this general knowledge to the specific case. Based on the general law, one can form a prediction for a specific instance and solve the problem based on a prediction of what is most likely to happen.

Induction means to develop general concepts and laws from examples. Of particular importance for making inferences are patterns, building a bridge between many specific instances and more general relationships (Margolis, 1987). For instance, in the decision science and artificial intelligence literatures, inductive learning refers to attempts to induce general concepts from examples by creating a decision-tree-like knowledge structure (Zhu et al., 2001). A classification procedure will often be involved

that models a known classification so that, when an object of an unknown class is encountered, a classification can be proposed for it (Tessmer, 1997).

The idea of abduction goes back at least to pragmatist philosopher Charles Sanders Peirce (1934). Recently, it has been argued in the engineering literature that an abductive reasoning process is the most important aspect of knowledge creation in designing new products (Baba and Nobeoka, 1998; Itoh 2003). Abduction refers to generating a consistent, and 'best possible' explanation (Peirce 1934; Baba and Nobeoka, 1998; Johnson and Krems 2001; Magnani 2001). Three aspects of abduction are important to take note of. First, abduction is seen as the sequential interpretation and integration of new data and hypotheses into a single explanation, in a cycle of theory building, formulating hypotheses, and testing hypotheses with experiments (Magnani 2001; Lapré and Van Wassenhove, 2001). That is, abduction is used to refer to the whole cycle. Second, abduction also carries a particular connotation of the hypothesis-generation step: while induction can also be used to generate hypotheses, abduction emphasizes the search for an underlying causal mechanism (Lawson, 1997). Where this search is successful, causal mechanisms are identified. The thrust of abduction and induction is therefore different, abduction being employed with an interest in identifying causal mechanisms. Applying abduction therefore has the consequence that the acquired knowledge can be justified (Lundberg 2000). Third, in generating hypothesis by abduction, one also draws on background knowledge, implicit assumptions, and the like. The hypothesis is therefore generated not exclusively on the basis of the given premises (such as in induction).

An important implication of the last characteristic is that abduction can be applied also when the problem at hand is unstructured, while both deduction and induction require the problem at hand to be structured. For induction, the minimum structural requirement is that the instances used to induce a regularity are in some respect similar to each other. For deduction, one needs to be able to map the different elements in the law to elements of the problem (Baba and Nobeoka, 1998; Pisano, 1994, 1996).

From the overview of the three problem-solving strategies, we note a difference in kind between abduction on the one hand, and in- and deduction on the other. Abduction means to *reflect on causal processes* that produced results and to *search for* and try to *identify the underlying causal mechanism*. Abduction, therefore, is like an additional layer that can be 'overlaying' learning-by-doing (inductive) or the application of cognitive regularities (such as if-then rules; deductive). In the case of induction as

implemented by learning by doing, abduction would mean that one applies induction *with the intention of and focus on identifying underlying mechanisms*. In the case of deduction, such as the application of general laws, abduction would mean to *reflect* on the rules that one applies, and to modify and adapt those according to feed-back from their application. The notion of 'double-loop learning' (Argyris and Schön, 1978) captures precisely this effect of abduction in deductive processes.

In the remainder of the paper learning and problem-solving, though closely related, will be considered as two distinct processes. Problem-solving is the wider notion. Problem-solving requires (acquiring) some knowledge to draw on, but the *application* of that knowledge to a specific problem is also a crucial part of problem-solving. Each time the knowledge is applied to a specific case, and there is feedback, knowledge is refined and added (i.e., learning takes place). But learning is not equal to problem-solving. In this paper, we focus on problem-solving, and thus on the application of knowledge to specific problems. Because many of the discussions that pertain to the issue in the focus here, however, centre on the term 'learning', we use these terms in order to provide a link to the literature. Our focus in this article is on physical (laboratory) and virtual (computer simulations) experimentation, because these provide feedback *during* the development process, not *after*, such as other learning forms (e.g. learning-by-doing in manufacturing).

3 Choosing between problem-solving strategies

Above, we have described three different problem-solving strategies, deduction, induction and abduction. What guidelines are available to choose between them? At least two arguments can provide guidance. Both build on Nelson's (1982) perspective of research and development activities as search activities, in which the knowledge about how to search efficiently for alternative solutions has an effect on the expected new product development success, measured in lead time, cost, and product quality (Wheelwright and Clark, 1992). The knowledge base therefore matters (Nelson, 1982; Levitt and March, 1988; Verganti, 1997). By a knowledge base we mean the knowledge that has to be drawn upon in order to design and produce the product in question (cf. Pisano 1994; 1996). In the case of a brake system, the knowledge base would include hydraulics, hydraulics and electronics, in the case of ABS, and so on.

The first argument that provides guidance focuses on the degree to which the knowledge base underlying the business activity (for instance, building cars), is formalized. A knowledge base is said to be formalized if it can be articulated, for

instance in the form of general laws such as the laws of physics. Knowledge bases that are not formalized are held in the form of experiential knowledge, heuristics, and the like. Bohn (1994) provides a proposal for how the formalization of the knowledge base could be operationalized. Where the knowledge in question is fully retained as tacit, implicit, experiential knowledge, one does not have any ‘hard-and-fast’, reliable and precise laws in order to apply deductive problem solving. Where a formalized knowledge base exists, the depth of knowledge about cause-effect relationships it affords will decide whether deductive or inductive problems-solving is the more appropriate choice. Pisano (1994) has argued that the structure of the knowledge base underlying the product in question plays an important role for deciding the scale of the set-up for learning: ‘In environments where prior knowledge is weak, high-fidelity feedback requires experiments in the actual production environment ('learning-by-doing'). In contrast, when reliable theoretical models and heuristics exist, laboratory experiments, simulation, and other forms of 'learning-before-doing' can be productively harnessed’ (Pisano, 1994: 86). A sufficiently strong scientific knowledge base means that effective learning may take place also outside the final use environment, for instance in laboratories ('learning-before-doing') (Pisano, 1994). Where the formalized knowledge base is not strong enough (for instance in biotechnology), 'high-fidelity feedback' is required which can be obtained only in the original production (or design) environment. The problem is to have sufficiently deep knowledge of the effect of specific variables and their interactions in order to simulate and predict effects 'off-line' (Pisano, 1994). Where that is possible, one can apply techniques such as computer simulations, laboratory experiments, prototype testing, pilot production runs, and other experiments for learning-before-doing (Pisano, 1996). In this sense, the formalized knowledge base permits learning through experiments performed in an artificial setting. The reason is that the formalized knowledge provides the context within which information gained in simulations and experiments is interpreted (Arora and Gambardella 1994).

The second dimension of the knowledge base which can provide an indication of which problem-solving mode to choose is the degree to which the knowledge base is rendered obsolete by developments in the environment. The most important of such developments, in most industries, is technological development. However, changes in customer demands, regulations, or wider economic conditions can also lead to making the existing knowledge base obsolete. Such a situation has become known as discontinuous change, which can be competence-destroying in the sense that a knowledge-base that previously provided the competences to operating successfully

with one core technology becomes obsolete and even a barrier to adopting and successfully operating the new technology (Abernathy & Clark, 1985; Henderson & Clark, 1990; Lynn, Morone & Paulson, 1996). One direct implication of such radical, discontinuous, competence-destroying (and knowledge-base devaluating) change is that firms need to turn to a different set of problem-solving mechanisms, also in order to build up a new knowledge-base. The problem is that due to change in the environment, the mental models that have mapped the environment do not fit any more. Therefore, deductive problem-solving will most likely not generate satisfactory results. Inductive problem-solving methods are the option of choice, including formal experimentation, rapid prototyping and improvisation. These latter are learning mechanisms that firms can utilize in such circumstances (Moorman & Miner, 1997). As we will elaborate below, problem-solving can be sped up by entering a double-loop problem-solving cycle which employs abduction.

These two knowledge dimensions, formalization of the knowledge base and the speed with which it changes, are linked. Formalization usually requires deep, not just partial understanding. Such deep understanding can rarely be achieved fast, but is usually the result of a long learning process. In particular where deep understanding includes, in some way or another, tacit elements that need to be acquired by implicit learning (Reber 1993), such processes take time because new cognitive categories need to be formed. It follows that when the knowledge base that needs to be drawn on in order to solve a problem in new product development changes fast (take biotechnology in developing new pharmaceuticals as an example), that knowledge base will usually not be available in formalized form. In Table 1, we propose that when knowledge is fully formalized it also encapsulates deep enough knowledge of the effect of specific variables and their interactions in order to predict effects 'off-line'. In this case, deduction problem solving strategies are the most effective and efficient choice.

When knowledge is not deep enough to predict by applying laws, on the other hand, there are two problem solving options. In the case in which the basic knowledge base is still valid and only some aspects of it have changed, one will be able to use abduction, relying on virtual reality and simulation to observe the implication of these changes. For example, in the computer simulation of a vehicle system, one can change the product architecture while still being able to predict the performances of the vehicle. On the contrary, if the materials and technologies employed change dramatically, the simulation rules will also change. In some cases, in highly dynamic technology environments, a firm may even lose the ability to formulate hypotheses on a given

material, product or technology performance and has to rely on inductive problem solving strategies.

Table 1. Choosing problem-solving strategies on the basis of the knowledge-bases required to perform the design task

Knowledge base dynamism	Low	High	Very high
Knowledge base formalization	Fully formalized: Encapsulates deep enough knowledge of the effect of specific variables and their interactions in order to predict effects 'off-line'	Not (yet) fully formalized: Does not encapsulate deep enough knowledge to predict by applying laws	Not (yet) formalized: Does not encapsulate deep enough knowledge to predict by applying laws
Capacity to formulate simulation principles	No need	Yes	No
Problem-solving strategy	Problem-solving by applying laws (deduction)	Virtual Simulation (abduction) (on the link between virtual simulation and abduction see section 5)	Physical Simulation (or some other inductive strategy)
Advantages	Cheap and very reliable when laws, norms, procedures are available or easy to acquire	Allows almost infinite iterations, thus enabling the search for causal relationships Because of that, faster (some learning steps can be bypassed), and leading to deeper understanding	Results are very reliable
Disadvantages	Only works when knowledge of the phenomenon exists	Not always reliable	Expensive and not able to identify clear causal relationships in complex products

4 Concrete problem solving mechanisms applied in new product development

When NPD activities allow for off-line problem solving, engineers can develop new design and engineering solutions simply by applying laws, design procedures and norms. The mechanisms at work in this case are those related to deduction. In order to reach the optimal design and engineering solution, however, engineers often need to go through several loops of induction and deduction (see figure 1). In reality, since experimentation can be applied both to physical and virtual objects, its nature is twofold. The distinction is subtle, but central for framing problem solving mechanisms and their applications.

Experimentation involving physical objects has some drawbacks. The cost and the time required to carry out this type of physical experiments is not always compatible with

strict NPD process time and cost constraints. Such constraints limit the use of experimentation, leading to the consequence that engineers have to rely on the “carry over” of components and system of components from the previous model generation to the next one. Conservative design, hence, is often the consequence of the impossibility to test new solutions fast and at a reasonable cost by way of on-line inductive learning and problem solving.

Virtual experimentation problem solving mechanisms try to overcome these limitations. The great calculation power reached by computers allows an almost infinite number of iterations of an experiment on a virtual object.

An example can be helpful for grasping this point. In the design of a car that could pass the NCAP crash test (http://www.euroncap.com) with a good mark, it is only possible to build a limited number of physical prototypes due to cost reasons. Given the fact that there are at least 400 components (interview with product development engineer, April 2003, see for further details Caputo, et. al. 2003) that affect such a crash test, it is impossible to isolate precise cause-effect relationships amongst all the parts (deductive problem-solving). Applying physical experiments, on the other hand (inductive problem-solving), would mean to produce several batches of car bodies, so that when reports about car crashes later come in, systematic observation allows drawing conclusions. Beyond the aforementioned cost limitations of this strategy, physical prototypes incorporate engineering solutions that, very often, by the time the prototype is crafted, are already obsolete. This makes the experiment results not fully relevant for the engineering development and, as a consequence, not only inefficient but also ineffective.

Finally, learning by way of virtual experimentation (abductive problem-solving, see section 5) would enable almost infinite iterations of the same experiment and, by isolating one parameter in each run, simulating a “laboratory-type controlled environment”. In this way, a great variety of hypotheses on the causal relationship between the design characteristics and the crash test performances can be tested. What is relevant here is that the simulation, working with (1) existing design rules, and (2) the knowledge based on engineers’ experience about the product and technology, helps to verify whether new design solutions are viable. In doing so, it also involves the engineers’ creativity. It is therefore not real-life experiences that provide the learning effect, but the testing of hypotheses. As a consequence, to the extent that it is possible to formulate cognitive models in such a way that they can be coded in simulation software, one can gain great benefits from applying virtual experimentation. The virtual

experimentation mechanism, hence, presents the ability to search for causal relationships. This is precisely what we have associated to abduction. Note that a first step in searching for causal relationships is narrowing down the problem space that needs to be searched. A high number of experiments – enabled by virtual experimentation – might be helpful here.

On the whole, virtual experimentation is the most appropriate tool to make abductive problem-solving viable. This is for three reasons. First, virtual tools, as shown, help engineers to *observe* phenomena which are not observable otherwise (see the crash test example, Thomke and Fujimoto 2000). Second, and most importantly, virtual experimentation allows testing non-conservative hypotheses, i.e. hypotheses that are not constrained by the logical bounds of the premises one starts from. It thereby enables non-conservative design, which is important in order to achieve distinctive new designs (Wheelwright and Clark 1992). Third, virtual experimentation is cost and time efficient in two ways: On the one hand, it is less costly and time-intensive than physical prototyping. On the other hand, it can prevent corrections in later stages, which are both costly and sources of delay (Thomke and Fujimoto 2000).

5 The case study

Not many studies have emphasized the role of abduction in new product development. Extant studies dealing with virtual reality and simulation focus on the applications and use of simulation (Murphy and Perera, 2002), on product development via immersive virtual tools (i.e. tools that allow the engineer to interact with the object of development as if it were physical) (Scharm and Breining, 1999, Bao et al., 2002), assembly (Jayaram et al. 1997), prototyping (Kelley, 2001), and production engineering (Klingstam and Gullander, 1999). These papers intend to contribute to the development of the simulation tools themselves, and only marginally deal with managerial implications for product development. Within the literature that frames virtual development as a tool for improving NPD performances, enabling problem solving and knowledge integration, the paper by Baba and Nobeoka (1998) is among the first to present a virtual simulation tool (3D-CAD) as an enabler of abductive problem solving. However, the field research was carried out some years ago (1995) and the technology has improved so much by now that a new analysis of the effects of Virtual Prototyping on abductive problem solving is needed.

D'Adderio's (2002) paper on virtual prototyping techniques, concepts and models, represents a first systematic attempt to analyse the knowledge integration properties of

computer-aided styling (CAS) and computer-aided design (CAD) tools. The paper provides an interesting analysis of the problems related to turning physical in virtual and vice versa, and of the organisational implications of using Digital Models. D'Adderio also stresses the role of "translation routines" in coordinating actions and knowledge across heterogeneous functions (and communities of practices).

Building on and extending this literature, we draw on a recent case study in the automotive industry. The study has been carried out by way of structured interviews with engineers of one of the Research Centres of a major European OEM, and industry experts at the Department of Mechanical Engineering of The University of Salerno. In the automotive industry, there has been a massive recourse to the problem solving tools we associate to abduction (Vasilash, 2000; Thomke, 2001b). A large number of different virtual simulation tools are currently applied in the auto industry (for further details see Caputo, Salvatore & Zirpoli, 2003). General Motors has created 19 centres for Virtual Reality (VR) applications in four different continents, making 20.000 engineers users of Computer Aided Design (CAD), Computer Aided Engineering (CAE), Computer Aided Manufacturing (CAM) tools and 17.000 users of Product Data Management (PDM) tools. At BMW, there are 7 Virtual Reality (VR) centres and thousands of work-stations for Virtual Prototyping & Simulation (VP&S) activities (see the following section for more details on Virtual Prototyping & Simulation (VP&S) activities). Development teams design cars and sub-systems through virtual simulation of about 20 first-level performances (Safety, Ergonomics, Performance & Fuel consumption, Emissions, Ride & Handling, Noise, Vibration, and Harshness (NVH), Aerodynamics, Thermal & Cooling, Durability, and so on) broken down into more than one hundred lower-level performances, each of them having its own numeric simulation. OEMs have also joined up with software houses specialized in numeric simulation, creating joint teams in order to develop software able to meet specific demand of product development process. One of the first results is the well-known car dynamics simulation software MSC.ADAMS/Car jointly developed by Mechanical Dynamics and a consortium of the world's most important OEMs, including Audi, BMW, Daimler-Chrysler, Nissan, Porsche, Renault, Rover, Samsung, Toyota, Volkswagen, and Volvo.

The research centre under investigation has recently begun to play a central role in the process of "virtualization" of the OEM NPD process. It counts around 750 employees and was established as a "green field" research centre close to one of the OEM's plants in 1988. The low average age of its engineers and their high standard university

training made the research centre an ideal place for the OEM where to invest in new technologies and methodologies for product development. The research centre is organised on the basis of seven functions: Engines, Vehicle, Technologies (production processes), ICT (information and communication technologies), Control Systems, Mobility and Road Safety, and Product Development Methodologies. Our field study involved managers from the Vehicle and Product Development Methodologies functions. They are, respectively, responsible for the development of the vehicle engineering (in particular all the mechanical systems except for car body and interiors) and for the design of new "methodologies" for the use of prototyping technologies in product and process development (in this paper we will refer mainly to this latter). The OEM and the Research Centre, in fact, re-arranged their organisation, in order to effectively involve the research centre in the OEM's products/processes development. This has required huge efforts in learning and developing new technologies, know-how and organisational solutions.

6 The use of virtual tools in practice

Specialized industry magazines and scientific literature report on the results of the virtual tools employment. Digital technologies have enabled OEMs to reduce Time To Market and development costs in order to adapt to shorter product life cycles. The results seem to be more important than any other revolution in the past, both regarding product development organisation and management. General Motors managed to reduce development time by more than 50%, from 40 to 18 months; Toyota achieved the same result, while BMW developed the X5 in 36 months instead of 60. These results were achieved replacing Physical Mock-Up cycles, which requires huge financial and time resources, with fast and cheap Digital Mock-Up (DMU) cycles. Developing a physical prototype takes about two months with costs between 250.000 and 400.000$, including physical tests planning and execution; on the contrary, a digital model for crash analysis takes less than one man-day and the whole crash simulation takes no more than two days, running even outside working hours, as the presence of the operator is not necessary. General Motors now performs only one Physical Mock-Up cycle, which it is compelled to by legal standards for consumer protection, while BMW performs two clay models: one for final style validation and one for the same reason as GM. GM saved 1 billion dollars in a year; in its three main platforms, Ford saved about 40 million dollars for CAE activities (including the investments necessary for virtual prototyping use) and over 1.000 million dollars thanks

to the reduction of design changes in late stages of its product development process. Further, Virtual Prototyping improved product design performances: between the introduction of virtual prototyping tools in the BMW NPD process, and 2002, the energy absorption degree in front crash tests increased by 2,4 times.

In the field study we did not investigate these performance figures, on which the literature is quite consistent, but have tried to systematise the adoption and implementation of virtual tools. (We have benchmarked the tools used by the OEM under investigation with those used by other OEMs, via industry reports or scientific literature and found a substantial overlap of tools, equipments and techniques used by the OEMs in the auto industry).

Results show that virtual development tools are of four types: Virtual Reality, Virtual Prototyping & Simulation, Virtual Factory, and Digital Mock-Up.

Virtual Reality (VR) studios provide digital *representation* (using wall-sized projection screens), taking the role earlier played by clay prototypes.

Digital Mock-up (DMU) is a complete digital product description for development, design and manufacturing. Most studies on Virtual Development focus on DMU.

Virtual Prototyping & Simulation (VP&S) tools consist of the mathematical representation of the laws that govern the physical phenomena involved in the behaviour of the vehicle and of the physical property of the systems that constitute the vehicle.

The Virtual Factory (VF) performs the same task of VP&S, but applied to manufacturing rather than product design.

The four simulation tools presented above can be clustered in two groups (Figure 2).

The first group, including *VR and DMU tools*, provides a digital *representation* that enables product developers to visualize the product and the process before its physical construction, including both the individual components as complete systems and their dynamic interactions. What distinguishes virtual reality (VR) from physical experimentation (using physical prototypes) is the possibility of real-time feedback by way of interaction between the prototype and the user. DMU is a prerequisite for adopting VR. In turn, it requires the use of computer aided development tools (CAx).

The second group of tools includes the *VP&S and VF tools*. What sets those apart from physical experimentation is that they enable *simulating* the interaction between the mathematical representation of the laws that govern the physical phenomena involved

in the behaviour of the vehicle and the physical property of the systems that constitute the vehicle, and thus the behaviour of the vehicle. This ability makes it possible to identify critical performance attributes, and to 'fine-tune' those without a physical prototype. VP&S removes current bottlenecks in the engineering process and enables concurrent engineering. Ultimately, it allows implementing a 'Design-Right-First-Time' approach.

It is important to note that the two groups of tools have a different impact on problem-solving. While the first group merely enables speed and cost improvements, VP&S and VF tools are able to alleviate the major limitation of physical experimentation, namely, the impossibility to observe the effect *of each individual parameter while holding all others constant* (*ceteris paribus*). This is due to the fact that each physical experiment will affect all the parameters at the same time. Virtual experimentation, on the other hand, is capable of testing a great variety of hypotheses on the causal relationship between the design characteristics and the crash test performances. The second group of simulation tools therefore opens up possibilities for *testing* hypotheses that simply did not exist without such tools. For this reason, they are associated to abduction. Figure 2 below reports the differences between the two groups of virtual tools.

Virtual Tool	**VR and DMU**	**VP&S and VF**
Task	Virtual *representation*	Virtual *simulation*
Scale	Any	From single component, to systems of components, to full products (in a tree structure)
Problem-solving strategy	Induction	Abduction

Fig. 2. Mapping virtual tools

We have just argued that the second group of simulation tools opens up possibilities for testing hypotheses that simply did not exist without such tools, making such simulation tools very attractive for product developers. So how can NPD managers most appropriately make use of these tools? Our field work indicates that in order to use VP&S & VF tools, a complex procedure is required. This will be the object of the next section.

6.1 How virtual tools are designed: the concept of virtual development routine

When the OEM requires a new virtual development tool, a customer-supplier type relationship between the research centre and the OEM is established. The OEM is the process-activating actor, having to satisfy product and process innovation requirements through the implementation of high technology- and virtual prototyping tools. Because

successfully applying virtual development tools requires the collection, and inputting of (all) the required data, as well as the capability to interpret the outputs of the virtual development tools and link them to other development tools, just acquiring a simulation software is not enough. At least as important, if not much more important, is systematic guidance of how to use the tool and deal with the inputs and outputs, as just described. This, too, is an important task of the Research Centre. It is of strategic importance that the OEM can use these technologies and tools through standard and formalised procedures, in order to achieve reliable results in terms of technologies/tools performances. This introduces a fundamental concept that we call virtual development routine (henceforth VDR) (in reality, the Research Centre uses to express this concept the term “methodology”. This term, in our opinion, can induce some misunderstanding on the nature of the object behind it). Virtual development routines allow the OEM to effectively apply the Research Centre virtual technologies/tools in its products/processes development. According to the definition provided by the Research Centre, VDRs have the role of (1) integrating design norms and procedures and of (2) enabling their application to virtual representations. More precisely, design norms specify “how” a certain object has to be designed and engineered, procedures specify the flux of activities to be performed and “who does what”. Design norms represent the codification of the company know how on how to perform a certain object design and are usually systematised during the development of the virtual tool. Procedures refer to the use of the tool itself once the product development team will implement it in practice.

The OEM, after having performed its market, scenario and benchmarking analyses, produces a request for a *virtual development routine* to simulate a particular “vehicle performance” (for example, a pedestrian crash or a frontal crash) to the Product Development Methodologies function of the research center. The fact that the research centre decided to set up a function dedicated to the development of virtual development routines shows how important such virtual development routines are, supporting our view that the use of virtual tools is much more complex than depicted in the literature and requires further investigation. Such a request means that a set of design rules and procedures, forming a *virtual development routine*, to input the virtual simulation tool has to be developed. The correct use (correct input procedure) and interpretation (correct analysis of the simulation results to predict the real behaviour of the vehicle) of the *virtual development routine* will allow a successful vehicle design that will achieve a certain reliability in the virtual experimentation of the “vehicle performance”. In other words, the *virtual development routine* will put the engineers in

the condition to use the simulation tool in a way that is reliable enough in simulating vehicle performances before real tests are carried out. Before the Research Centre (Product Development Methodologies function) starts the development of the simulation tool, it scouts the simulation technologies available on the market (software development is always outsourced). The *virtual development routine* is developed by mathematically modelling the behaviour of a “real vehicle” (a vehicle which is already on the market) on which it is possible to test the real performances. The input data on which the *virtual development routine* is built, hence, are “real” design data coming from an existing vehicle. The first release of the *virtual development routine* is tested comparing the results of the “vehicle performance” performed on the “real vehicle” in a “full scale” simulation and the output of the virtual simulation based on the same input data. The *virtual development routine* is refined on the basis of the comparison between the results obtained from the virtual simulation and the results obtained by the physical experimentation on the real vehicle. It is worth noting that the virtual tool reliability has to approximate 100% in order to get a first validation. At this stage there is a first release of the *virtual development routine* which allows reliable simulations on the vehicle taken as the object of the test (data coming from virtual and real experimentation converge). The problem at this stage is that the *virtual development routine* is reliable for an *a posteriori* validation of the vehicle that is already on the market and is not necessarily able to predict the performance of a completely new vehicle. In other words, the *virtual development routine* has to prove itself to be generally applicable. The final validation of the *virtual development routine*, hence, is released only when the new *virtual development routine* is tested on another real case. Usually, the new comparison is performed with a second model which is already on the market, which is considerably different from the one on which the *virtual development routine* has been previously defined. The *virtual development routine* is validated when it allows generating simulation results that are reliable and general enough to be applied to simulate future product behaviour with respect to “vehicle performance”: an acceptable threshold for this purpose is a +/- 3% gap between simulation output and real performance.

The *virtual development routine* development just described may last from 12 to 24 months, varying according to the complexity of the performance to be simulated. As far as the people involved in the *virtual development routine* development are concerned, the Research Centre needs to involve engineers coming from the OEM to learn about the components or systems that will be the object of the simulation. The people coming from the OEM will act as “gate keeper” in so much that they will be able, once the

virtual development routine has been defined by the Research Centre to apply it in the next generations of NPD. On the other hand, it may sometimes happen that somebody from the Research Centre is co-located at the OEM to develop the new virtual development routine there. It is worth noting that a good virtual development routine should take into account both design and manufacturability issues and, as exemplified define new design and engineering norms.

6.2 How virtual tools are implemented

Once a *virtual development routine* is defined it should provide a flexible simulation procedure in order to evaluate specific car performances and find out eventual design faults. The *virtual development routine* shows the user (the OEM development team) how to use virtual simulation tools, anticipating information previously obtained only after physical experimentation.

Virtual development routines are used in the early stages of product development by both members of vehicle development teams and members of functional units of the OEM who then provide simulation results to vehicle development teams (these phases can be framed approximately between -60 and -40 months before vehicle launch on the market). The full range of *virtual development routines* should realize the coordination necessary to codify information on the various components and systems of the vehicle into a digital format that, through the company PDM (Product Data Management System), should be the input data for all the simulation activities during the early phases of vehicle design (for example, the CAE and CAM teams should use the same design input, stored in the PDM, to perform their simulations).

It is interesting to note that the *virtual development routine* is developed in order to allow the designers to predict the behaviour of the vehicle before physical tests are performed. This result, however, is achieved by testing the *virtual development routine* on two real cases in which not only the input of the routine is available (the design coordinates) but also the behaviour (physical experimentation). Together with the simulation tool, the new *virtual development routine* provides: (i) a new procedure (that formalises the structure of the design process in order to gather robust and consistent results during the virtual simulation of the “object” performances) (ii) new calculus norms, and (iii) the error rate which is considered acceptable when simulating (i.e., the gap between the output of the simulation and what will happen in reality).

A relevant aspect of the *virtual development routine* is that the development and the use of the virtual development tools require the definition of these routines that allow

the development team to benefit from software simulation tools. These are often general software applications that need to be customised and structured in a way that all development functions involved can use them. This activity is a complex one and requires the convergence of many bodies of knowledge. The *virtual development routine* development teams, in fact, are formed by (i) staff who contributes with its know how on the product and component technologies (very often a representative of the OEM), and (ii) staff who is knowledgeable on (1) the type of performance that has to be simulated (in the example he or she is informed on the characteristics and performances of pedestrian crashes), (2) the laws that regulate the interactions between materials and components (very often the unit which develops the *virtual development routine*), and (3) the software and hardware technology (in our field work, a representative of the Research Centre).

7 Discussion and Conclusion

Applying a problem-solving perspective and investigating the different problem solving modes available, we have first identified the advantages of problem-solving by way of *abduction*, as compared to *induction* and *deduction*, and presented a rationale of how to decide between the different problem-solving strategies; based on a case study of a major European OEM in the car industry, we have then analyzed how problem-solving by abduction is implemented in practice and argued that virtual experimentation is the tool most appropriate to implement *abductive* problem-solving. Several considerations arise from our analysis.

The first observation concerns the way in which simulation tools are perceived in new product development research. While the paper by Baba and Nobeoka (1998) has been important in breaching the topic, our analysis indicates that the simulation tools dealt with in their paper cover only one of two groups of simulation tools. As figure 2 shows, the group of tools they deal with is the less powerful one regarding abductive problem-solving. Rather, they describe what are (at least in 2003, if not in 1995 when their research was carried out) very basic building blocks of virtual *representation* tools. As drawing-boards are largely out of use nowadays, 3D-CAD is just a basic tool used by engineers for representation, just like a word-processor is used as a tool for writing. 3D-CAD does not, however, allow to simulate, and thus apply abductive problem-solving (except for parametric CAD software for product mathematics generation).

We can use this example to underline the second result of our analysis, namely having identified why precisely the group of simulation tools we have termed Virtual

Prototyping & Simulation (VP&S) and Virtual Factory (VF) enable and foster abductive problem-solving. First, virtual experimentation opens the possibility to isolate individual cause-effect relationships by varying individual factors and holding all others constant. On such a high level of sophistication, such a quality of experimentation is possible only with the group of virtual simulation tools that we have termed Virtual Prototyping & Simulation (VP&S) and Virtual Factory (VF). As a consequence, virtual development tools do not only boost NPD performances but also deeply change the nature of problem solving. In particular, abductive problem-solving is an emergent and relevant problem-solving mode which offers advantages over inductive and deductive problem-solving. In fact, the observability of causal relationships is much higher than in the other modes of physical and virtual experimentation (in physical crash tests, it is very difficult to observe the impact of the crash on many of the internal parts of the car). That, too, makes it much easier to identify causal relationships.

A third contribution of the article has been to systematise and frame problem solving strategies (Table 1), thereby providing managers with a tool to choose among them. The most important implication of this, corroborated by our empirical findings, is that virtual tools and experimentation cannot completely substitute for the physical experimentation on which they build.

As a consequence, the nature of the process of abduction itself is also clarified further. As we have pointed out, abduction means to reflect on the causal processes that produced results. Abduction is thus like an additional layer that can be ‘overlaying’ inductive or deductive problem-solving. It introduces the meta-level of reflecting and generating hypotheses, which are then deliberately put to test (figure 3).

For an illustration of what it means to say that abduction is on a ‘meta-level’, consider the case description of how the *virtual development routine* was developed. It was developed based on two kinds of knowledge inputs: the knowledge about physical laws (on part of those who develop the specifications for the programmers), and observation of the physical experimentation. Remember that once a *virtual development routine* is developed, the results of the virtual simulation are compared to the results of the physical simulation, both using a car model that is already in the stage of mass production. Only due to the fact that *both* kinds of knowledge are used for triangulation, *and* that there is reflection about the differences, the team was able to produce a reliable *virtual development routine* for future models under development. Furthermore, note the importance of learning from successive generations of car models for achieving this result.

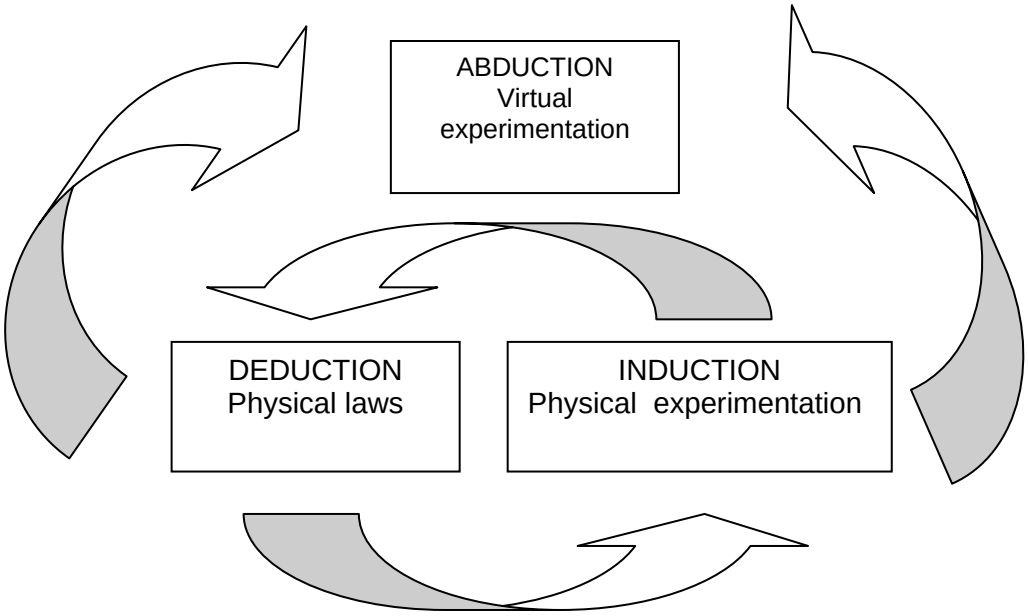

Fig. 3. The modified problem-solving circle

Finally, from an organisational point of view, there are interesting insights into the nature of teams used in abductive problem-solving. Interestingly, the team that develops virtual simulation methodologies was staffed on the one hand by engineers belonging to the research center, who both have knowledge of software and of physical laws. On the other hand, the team was staffed by engineers from the OEM that have product knowledge. Those, however, do not come from the NPD teams, but rather from the engineering functions (in this case, the engineering divisions of the OEM). The organization structure at the heart of abductive problem-solving in this case was neither a function, nor a project in the usual sense (i.e., projects having deep product knowledge). Rather, it was formed by people from all the functions (not from project structures) who simultaneously act as gatekeepers (since they go back to their function once the *virtual development routine* has been defined). The criteria with which these teams are defined are functional (i.e. a team that develop a *virtual development routine* to design cars that perform a certain mark at the NCAP TEST is formed with specialist of components that affect this performance) and more specifically is based on the performance (i.e. *virtual development routine* to test “pedestrian crash”, *virtual development routine* to test noise perceived by passengers, etc). This means that the *virtual development routine* development teams are neither pure functional nor pure project oriented (in fact, people from NPD teams are never involved). This seems to be an important hint at how to organise abductive problem-solving.

In conclusion, virtual tools and the abductive problem-solving modes they foster make a great contribution to the work carried out in complex product design and industrialisation. They boost NPD efficiency and effectiveness overcoming the limits of problem solving based merely on induction and deduction, rooted respectively in experiential knowledge and theoretical knowledge. Most importantly, they allow non conservative solutions and to deepen the knowledge of the problem at hand. Understanding how virtual tools are implemented in practice and their contribution to changing the nature of problem solving and, consequently, the structure and organisation of NPD activities is a fruitful way ahead for research and practitioners alike.

References

Abernathy, William J. and Kim B. Clark (1985): Innovation: Mapping the winds of creative destruction. Research Policy, Vol. 14, 3-22.

Argyris, Chris and Donald A. Schön (1978): Organizational learning: A theory of action perspective. Reading, Mass.: Addison-Wesley.

Arora, Ashish and Alfonso Gambardella (1994): The changing technology of technological change: general and abstract knowledge and the division of innovative labour. Research Policy, Vol. 23, 523-532.

Baba, Yasunori and Kentaro Nobeoka (1998): Towards knowledge-based product development: the 3-D CAD model of knowledge creation. Research Policy, Vol. 26, No. 6, 643-659.

Bao, J.S., Y. Jin, M.Q: Gu, J.Q. Yan and D.Z. Ma (2002): Immersive virtual product development. Journal of Materials Processing Technology, Vol. 129, 592-596.

Bohn, Roger E. (1994): Measuring and Managing Technological Knowledge. MIT Sloan Management Review, Vol. 36, No. 1, 61-73.

Caputo, Mauro, P. Salvatore and F. Zirpoli (2003): Opportunities And Threats Of Computer Based Product Development, 10th International Product Development Management Conference, Brussels - Belgium, June 9-11, 2003,

D'Adderio, Luciana (2001): Crafting the virtual prototype: how firms integrate knowledge and capabilities across organisational boundaries. Research Policy, Vol. 30, 1409-1424.

Gavetti, Giovanni and Daniel Levinthal (2000): Looking Forward and Looking Backward: Cognitive and Experiential Search. Administrative Science Quarterly, Vol. 45, 113-137.

Henderson, Rebecca M. and Kim B. Clark (1990): Architectural Innovation: The Reconfiguration of Existing Product Technologies and the Failure of Established Firms. Adminisrtative Science Quarterly, Vol. 35, 9-30.

Iansiti, Marco (1995): Technology integration: Managing technological evolution in a complex environment. Research Policy, Vol. 24, 521-542.

Iansiti, Marco and Kim B. Clark (1993): Integration and dynamic capability: Evidence from product development in automobiles and mainframe computers. Harvard Business School Working Paper No. 9304 7.

Itoh, Toshio (2003): Abduction for Creativity. International Journal of Technology Management, Vol. 25, No. 6/7, 507-516.

Jayaram, Sankar, Hugh I. Connacher and Kevin W. Lyons (1997): Virtual assembly using virtual reality techniques. Computer-Aided Design, Vol. 29, No. 8, 575-584.

Johnson, Todd R. and Josef. F. Krems (2001): Use of current explanations in multicausal abductive reasoning. Cognitive Science, Vol. 25, 903-939.

Kelley, Tom (2001): Prototyping is the shorthand of innovation. Design Management Journal, Summer 2001, 35-43.

Klingstam, Pär and Per Gullander (1999): Overview of simulation tools for computer-aided production engineering. Computers in Industry, Vol. 38, 173-186.

Lapré, Michael A. And Luk N. Van Wassenhove (2001): Creating and Transferring Knowledge for Productivity Improvement in Factories. Management Science, Vol. 47, No. 10, 1311-1325.

Lawson, Tony (1997): Economics and Reality. Routledge, London.

Levitt, Barbara and James March (1988): Organizational Learning. Annual Review of Sociology, Vol. 14, 319-340.

Lundberg, C. Gustav (2000): Made sense and remembered sense: Sensemaking through abduction. Journal of Economic Psychology, Vol. 21, 691-709.

Lynn, Gary S., Joseph G. Morone & Albert S. Paulson (1996): "Marketing and Discontinuous Innovation: The Probe and Learn Process", California Management Review, Vol. 38, No. 3, 353-375.

Magnani, Lorenzo (2001): Abduction, Reason, and Science – Processes of Discovery and Explanation. Kluwer Academic / Plenum Publishers, New York.

Margolis, Howard (1987): Patterns, Thinking, and Cognition – A Theory of Judgement. University of Chicago Press, Chicago.

McDonough, Edward III, and Gloria Barczak (1992): The effects of cognitive problem-solving orientation and technological familiarity on faster new product development. Journal of Product Innovation Management, Vol. 9, 44-52.

Miner, Anne S., Paula Bassoff and Christine Moorman (2001): Organizational Improvisation and Learning: A Field Study. Administrative Science Quarterly, Vol. 46, 304-337.

Moorman, Christine and Anne S. Miner (1997): The Impact of Organizational Memory on New Product Performance and Creativity. Journal of Marketing Research, Vol. 34, 91-106.

Murphy, C.A. and T. Perera (2002): The definition of simulation and its role within an aerospace company. Simulation Practice and Theory, Vol. 9, 273-291.

Nelson, Richard R. (1982): The Role of Knolwedge in R&D Efficiency. Quarterly Journal of Economics, Vol. 97, No. 3, 453-470.

Nightingale, Paul (1998). A cognitive model of innovation. Research Policy, Vol 27, 689-709

Nightingale, Paul (2000): The product-process-organisation relationship in complex development projects. Research Policy, Vol. 29, 913-930.

Peirce, Charles Sanders (1934): Collected Papers of Charles Sanders Peirce. C. Hartshorne and P. Weiss (eds.). Harvard University Press, Cambridge/MA.

Pisano, Gary P. (1994): Knowledge, Integration, and the Locus of Learning: An Empirical Analysis of Process Development. Strategic Management Journal, Vol. 15, 85-100.

Pisano, Gary P. (1996): Learning-before-doing in the development of new process technology. Research Policy, Vol. 25, 1097-1119.

Tessmer, Antoinette Canart (1997): What to learn from near misses: An inductive learning approach to credit risk assessment. Decision Sciences, Vol. 28, No.1,105-120.

Thomke, Stefan H. (1998a): Simulation, learning and R&D performance: Evidence from automotive development. Research policy, Vol. 27, 55-74.

Thomke, Stefan H. (1998b): Managing Experimentation in the Design of New Products. Management Science, Vol. 33, No. 6, 743-762.

Thomke, Stefan and Takahiro Fujimoto (2000): The Effect of 'Front-Loading' Problem-Solving on Product Development Performance. Journal of Product Innovation Management, Vol. 17, 128-142.

Thomke, Stefan H. (2001a), "Enlightened experimentation: The new imperative for innovation**",** Harvard Business Review, Vol. 79, Iss. 2, pg. 67.

Thomke, Stefan H. (2001b): Managing digital design at BMW. Design Management Journal, Vol. 12, No. 2, 20-30.

Verganti, Roberto (1997): Leveraging on systemic learning to manage the early phases of product innovation projects. R&D Management, Vol. 27, No. 4, 377-392.

Wheelwright, Steven C. and Clark, Kim B. (1992): Revolutionizing Product Development. The Free Press, New York.

Zhu, Dan, G. Premkumar, Xiaoning Zhang and Chao-Hsien Chu (2001): Data mining for network intrusion detection: A comparison of alternative methods. Decision Sciences, Vol. 32, No. 4, 635-660.