Software is Orgware – A Semiotic Perspective on Computer Artifacts

Peter Brödner
Institute for Work and Technology
Munscheidstr. 14, D - 45886 Gelsenkirchen
Peter.Broedner@t-online.de

ABSTRACT
Contrary to common belief, IT systems often disappoint the expectations to increase productivity and flexibility of work and value creation processes. Moreover, most IT design and implementation projects still fail or burst time and cost budgets to a high extent. After presenting significant empirical evidence for these phenomena, the paper reflects on the reasons for their persistence by developing a semiotic perspective on the processes of dealing with computer artifacts in organisations. This semiotic view allows to understand these processes of designing, implementing and using IT systems as efforts of structuring social practices in organisations. Finally, a number of guidelines for an improved practice of designing and appropriating IT systems for effective use in organisations are derived from these theoretical reflections.

Author Keywords
Software crisis, IT productivity paradox, semiotic perspective on computer artifacts, computers as means of organising.

INTRODUCTION

Information Technology (IT) has often been characterised as “enabling technology” connected with far reaching promises. IT should allow for new forms of work organisation, open up new ways of organising value creation processes or even provide opportunities to create new businesses. Moreover, it should, according to common belief, lay the ground as a basic general-purpose technology for doing work more effectively and efficiently in a flexible environment.

Some of these promises have doubtlessly come true. However, most real IT implementations have turned out to be a barrier to rather than an enabler for organising flexible and more productive work and value creation processes. In essence, there are two strongly investigated empirical indicators for the unfulfilled promises and disappointed expectations: the so-called IT productivity paradox and the persistence of the software crisis.

Although there are growing bodies of empirical evidence for both phenomena, they are widely neglected in practice. In contrast, this paper wants to take the empirical evidence seriously and intends to reflect on the reasons for it. Why is it that only so few organisations succeed to substantially improve their economic performance by the use of IT systems? What are the reasons for the fact that, after forty years of strong software engineering efforts, still so many IT development and implementation projects fail again and again?

To this end, the paper starts with some significant empirical findings for both phenomena. It then develops a theoretical perspective on the nature of computer artifacts and their use for two reasons: First it can explain the empirical findings and second it serves as basis of cognition from which a number of guidelines can be derived for an improved practice of designing, implementing and using IT systems.

PERMANENT SOFTWARE CRISIS AND PRODUCTIVITY PARADOX: EMPIRICAL EVIDENCE

The Persistence of the Software Crisis

On a famous NATO Conference in 1968, the software crisis has been analysed and declared for the first time. Twenty-three years later, in 1991, Mitchell Kapor, the founder of Lotus Development Corp., stated in his Software Design Manifesto: “The lack of usability of software and poor design of programs is the secret shame of the industry” [11: 3]. And in 2004, another thirteen years later, a high level expert group in the UK put forward still again basically the same complaints about extraordinary high failure rates in the software industry culminating in the paradox: “We know why projects fail, we know how to prevent their failure – so why do they still fail?” [27: 10].

As a matter of fact, IT application projects do completely fail or at least burst their cost and time budgets to an extent and frequency which is markedly higher than in classical engineering disciplines. This is impressively confirmed by empirical findings from the Standish Group whose regular investigations collect data from large numbers of software application projects. In a recent survey from 2001 [32], based on data from over 30.000 projects, they found that only slightly more than one quarter of the projects succeeded, i.e. that they were completed on time and on budget, with all features and functions originally specified. All other projects either failed completely (cancelled before completion) or were challenged, i.e. completed and operational, but over-budget, over time estimate and with less functions than initially specified. And this did not substantially change over time (Fig. 1).

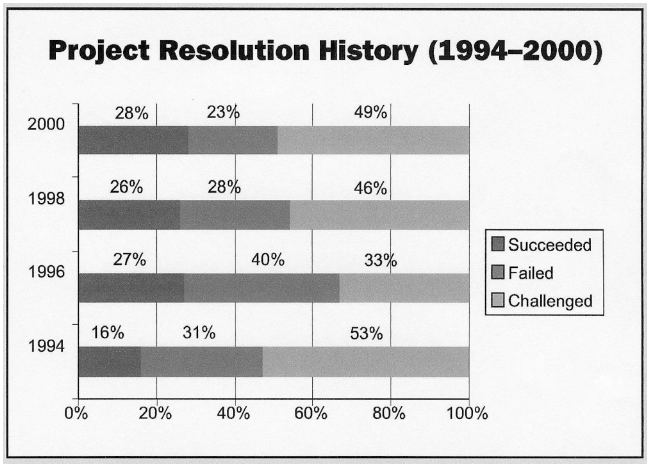

Figure 1: Success and failure of IT application projects (Standish Group)

From another empirical study on software failures we further know that the probability for failure highly depends on the size and complexity of the IT application projects. This probability grows exponentially with size up to a 50% cancellation probability for large projects with over 10.000 software functions [9].

Those complete software failures that have become known to the public, of course, form a peak of an iceberg only, since most failures remain hidden. However, the few that have been analysed all point, despite big differences between them, to the same reasons for failure again and again (see e.g. [24, 28]): insufficient project management and project controlling, underestimated complexity, lack of communication between designers and users, frequent changes of requirements during design and implementation, delayed decisions for progress, and incomplete documentation. Only recently, this has again been confirmed by the Royal Academy of Engineers: Alarming numbers of IT application projects “fail to deliver key benefits on time and to target cost and specification. ... This can be ascribed to general absence of collective professionalism in the IT industry, as well as inadequacies in the education and training of customer and supplier staff at all levels” [27: 4].

The IT Productivity Paradox

In accordance with these observations, we find the widely investigated so-called IT productivity paradox according to which IT often fails to increase productivity (for an overview cf. [2, 12]). Despite huge and ever growing investments in IT over decades, no noticeable additional productivity effects have been observed on the macro level of the economy. In the USA e.g., real annual IT investments have increased by more than ten times from a level of 20 Billion USD in 1975 to a level of 220 billion USD in 1990. In the same period of time, productivity in manufacturing has increased by the same small average annual growth rates as before while productivity in the non-manufacturing sectors has even stagnated [3].

This has not changed so much since, although productivity in the USA – where investments in IT regularly surpass those in manufacturing technology since 1991 – has significantly increased in the second half of the 1990ies from an average annual growth rate of 1% in the years 1987-1994 up to an average annual growth rate of almost 2,5% in the period between 1995-2000. Many observers have ascribed this productivity growth to IT. However, as the most recent productivity study analyses, this extraordinary productivity leap was solely caused by specific and unique developments in just six sectors: wholesale and retail trades, security and commodity brokers, electronic and electric equipment, industrial machinery and equipment, and telecom services. Surprisingly, these unique developments mainly deal with organisational redesign of the value chains rather than higher efforts in IT system implementations [15].

Since productivity investigations on the macro level are admittedly problematic due to a number of measuring problems and to possible compensating effects of a multitude of simultaneous changes, the focus of interest in studying the paradox has switched to the micro level of firm performance. Firm level investigations have indeed produced a number of remarkable results. Besides a great number of case studies, econometric analysis of data from ca. 400 big US companies [4] points out that

IT systems may improve the economic performance of companies, if and only if their implementation goes hand in hand with decentralisation, object-oriented reorganisation of work and investment in human capital,

“intangible assets”, e.g. collective action competence, strongly influence the benefit of IT systems,

companies decentralising their organisational structures achieve higher productivity in using IT systems than those who invest in IT only,

the expenses for organisational renewal and training are a multiple of the expenses for hard- and software, e.g. four times higher in case of implementing ERP systems.

Our own research on the implementation and use of ERP systems in German manufacturing enterprises produced comparable findings. Seven out of ten companies follow a purely technology-centered strategy and a top-down system implementation procedure with highly detrimental consequences for their economic performance. Thus, IT implementation projects regularly burst time and cost budgets to a considerable extent, while relevant performance indicators such as productivity, lead-time and in-process inventories are hardly improved, despite the extremely high expenses. The implementation process mainly concentrates on requirements engineering and design issues without end user participation, and efforts for appropriation and training are low. As a consequence, many functions of the system are not or poorly used, necessary knowledge about the integration in underlying business processes, their working principles and conditions is lacking, and large amounts of deficient or redundant data are being produced in use.

A small minority of firms only follows a more sophisticated and economically much more advantageous strategy starting with organisational redesign of their business processes and object oriented reorganisation of work with a clear customer focus. With these new organisational structures in mind, they simultaneously implement the functionally adapted IT system as a supporting tool and medium for cooperation. Accordingly, end users are strongly involved in these processes of organisational design and system implementation from the beginning and collective learning processes for appropriating and enacting the new ways of working are systematically organised [2, 13,14].

Similar findings have also been reported from case studies by other researchers [5, 6]. They obviously point to what is behind the paradox: How organisations understand and deal with computer artifacts either as means to automate existing work or as enabling and supportive media for creating and enacting an improved organisational practice decides about the economic benefits that can be gained. Making effective and beneficial use of computer artifacts is obviously more than implementing a functionally appropriate system.

THEORETICAL REFLECTIONS: COMPUTERS AS SEMIOTIC MACHINES

Semiotic Analysis of IT Systems: A Necessity

The misery indicated by these empirical data is, among other things, deeply rooted in conceptual deficiencies. So far mainstream computing science has – to some degree with the exception of the Scandinavian school – treated computer artifacts in much the same way as traditional engineering disciplines have treated their artifacts: By analysing relevant processes, functional specifications could be derived which the envisaged machine then had, as the result of a design process, to comply with. However, computers are symbolic machines manipulating data that represent information; their working principles obviously are fundamentally different from devices transforming energy or matter. Unfortunately, computing science has failed so far to develop an appropriate conceptual understanding of information or sign processes in which computers are embedded. Instead, the discipline has, besides its physical and mathematical foundations, strongly elaborated its requirements engineering and design methodology, but more of the same remedy only produces more of the same misery.

Sign processes, however, are a ubiquitous phenomenon: “Through almost all our life we are treating things as signs” [20]. The creation and use of signs as well as the treatment of information and meaning clearly are results of social interaction and, hence, their analysis falls into the domain of sociology. Unfortunately, the realm of things, how people conceive, sensibly act and interact with the objects they deal with in everyday life, reversely is being almost neglected in modern sociology. As a result, the comprehension of how people make sense of their artifacts in use, in particular computer artifacts, is poorly developed in sociology. As some kind of symmetric ignorance, both conceptual deficits, the lack of understanding sign processes and information in computing science as well as the missing comprehension of human interaction with technical artifacts in sociology, can at least partially be made responsible for the misery of inappropriate design and unproductive use of computer artifacts. Consequently, conceptual considerations must start to deal with these deficits. Those presented here are based on the pragmatic tradition of thinking, namely on the concept of sign by C.S. Peirce and the comprehension of things by G.H. Mead.

Signs are, according to Peirce, objects or processes that, in the view of an interpreter, stand for other objects or processes: A sign is “standing for something to someone in some respect”. Signs are our windows to reality, without them we could not even perceive it or sensibly act within it. In this perspective, a *sign* is a triadic relation (I–>(R–>O)) between three entities: (1) the representamen R as a material substrate of the sign (the object being interpreted as sign), (2) the designated object O and (3) the interpretant I as the meaning being assigned to the pair (R,O) through interpretation [25]. This sign concept is *recursive*: The interpretation is itself a sign that can be interpreted again.

In this perspective, computers can be identified as semiotic machines forming an own class of machines that can be well distinguished from the class of machines transforming energy or matter [1]. In the first instance, both types of machines have in common their close relationship with language, since they incorporate intentionally designed functions on the basis of concept formation and explicit knowledge. Humans have to interpret these functions within their action context in order to make sensible use of them (the functional “language” of the artifacts). The effects produced by these well-defined functions are then solely determined by the inputs. In order to make sensible inputs, intended use actions must be expressed in the functional language of the artifacts. This holds for all technical artifacts, from the hand-axe to the computer.

The fundamental differences between both classes of machines, however, lie in their domains of operation, their working principles and their purposes. The *operational domain* of energy or matter transforming machines as well as of chemical or biological artificial processes lies in nature as they purposefully intervene in natural processes transforming energy or matter, while the operational domain of semiotic machines is completely embedded in the social space of human interaction as they aim at converting signals or data within related sign processes. The processing of semiotic machines does not leave the social space of sign processes and meaningful interaction at all. Accordingly, the *working principles* of energy or matter transforming machines are completely based on natural effects as perceived by knowledge and their *purpose* is to make use of natural forces. The working principles of semiotic machines, by contrast, are based on acting instructions derived from explicit modeling of sign or interaction processes and their purpose is to organise and coordinate collective acting.

According to these distinctions, the interpretatory flexibility in dealing with energy or matter transforming machines is bound to and constrained by natural conditions, while in case of semiotic machines it is based on habits and conventions that themselves are affected by the models and instructions implemented in the machines. Consequently, their design and use face all problems of "double hermeneutics" present in sign processes of social systems [10]. In particular, the practice of dealing with semiotic machines in organisations[8] needs to be based on the development of a sufficiently shared information space and frame of interpretation [22].

Signals and Signs: From Physics to Semantics

Signs being used for computer processing can be specified as "algorithmic signs" [17, 18]: As precise analysis reveals, the use of computers in organisations is based on two coupled sign processes interlinked by the same representamen. While interacting with the computer, humans use signs as input that are meaningful to them in their action context. Inside the IT system, these signs, being readable and meaningfully interpretable in the outside context, are reduced to pure electronic signals as their material substrate. The signals don't "know" any more for what they stand. Rather, they are being processed through a program according to the completely determined instructions of the underlying algorithm. In Peircean notation the algorithmic instructions in this sign process reduced to syntactical operations on signals take the role of an interpretant, however a "causal interpretant" that formally falls in one with the designated object (Fig. 2).

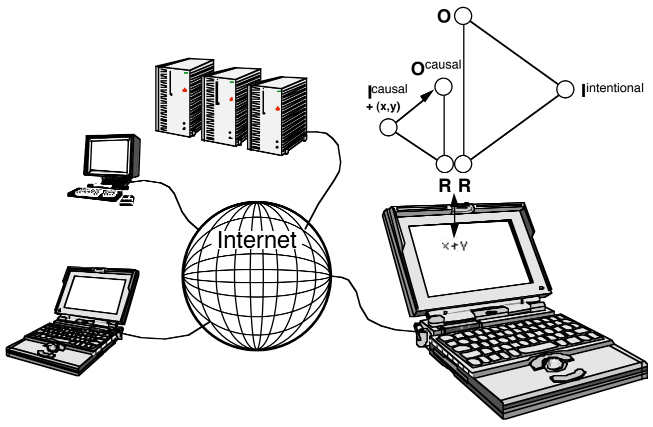

Figure 2: "Algorithmic sign" [18]: Unity of internal signal and external sign

The completely determined result of these syntactical operations on signals can, as its representamen appears on the interface, be interpreted again as sign within the social space of the action context. Consequently, computer-mediated social interaction is internally characterised by causal determination ("causal interpretant") of signal processing and externally by sense making interpretation ("intentional interpretant") of the signs associated with the signals. Inside the semiotic machine we find the effects of pure semiconductor physics and formal logic, while the events of social interaction outside are determined by semantics, the assignment of meaning in human action. Consequently, the social space of sign processes in interaction has not been deserted at any time. Rather, certain aspects of social interaction are being modeled within the computer system as a sequence of program instructions or "auto-operational form" [7]. Hence, the semiotic machine can also serve as a medium of organising sign processes.

This perspective discloses the semiotic nature of software: It exists as a finite description in form of a program text, that in turn determines, as operational code, a set of sequences of signal states of the hardware. These signal processes can, as they are embedded in human action contexts, be purposefully designed and meaningfully interpreted. Accordingly, software is double-faced in nature: It is (however awkwardly) readable text on one hand, and executable operational code, i.e. a machine, on the other. This exactly is a remarkable difference to descriptions of traditional machines (drawings and parts lists) that cannot directly execute themselves as machines. As a consequence of the semiotic nature of software, its usability, irrespective of its correctness, cannot be evaluated but in the users' action context.

According to Mead, even exploratory and instrumental acting in dealing with things is of social nature: Things do exist only so far as they also exist for others. Through our intentional relationship to the world around us as well as enabled by the action competence developed through

[8] The class of semiotic machines can be further divided into the subclasses of organisational systems and embedded systems. The latter serve as control devices for natural processes or machines in which they are embedded; they are not considered in this paper.

socialisation and previous acting, we are able to assign meaning to things or events we encounter. By exploratory acting with them, we conceive their functions and comprehend how we can use them intentionally and purposefully. By remembering the action schemes and their recurring characteristics, we form classes or concepts of objects or events in the outside world. By acting and interacting with others in a shared world, we "create" the things and ourselves, seeing them as taken for granted [16].

Mental reflections on our acting and its conditions are caused only, if hindrances or surprises occur in the flow of acting. Such action problems lead to a situation in which the things taken for granted are losing their "objectivity", since objectivity is not naturally given, but ascribed through shared understanding. Obstacles in acting trigger a reflection and search process in order to re-establish the "vanished object" and to regain the capacity to act (cf. the notions of "break-down" and "reflection-in-action" with Schon [29]). However, the experienced disorientation in such acting crises not only relates to the object, but also concerns the acting person itself. In the moment of uncertainty not only the world outside, but also the own power of judgment is being questioned. The acting person is unable to "distinguish between subject and predicate": "I want to emphasise that, as long as we don't have a predicate, we also don't have a subject" [16]. Nevertheless, through such processes of reflecting we can regain the capacity of fluid acting. This capacity includes the ability to anticipate the functions and properties of things learned from previous actions and to organise own actions according to the anticipated "thing behaviour" (cf. the notion of "situated action" with Suchman [30]).

These considerations apparently are also closely related to a similar approach based on the cultural-historic school of thinking, namely on activity theory, as proposed by Taxén in analysing computer-aided collective design processes of complex technical artifacts (Taxén's paper in this volume; cf. [19, 31]).

Software is Orgware

By virtue of the Peircean concept of sign and the Meadan comprehension of dealing with things one gets seamless access to modern theories of social systems that mediate between the views of subjective acting and objective acting structures and that can, in particular, appropriately explain both inertia and dynamics of collective acting in organisations. In this theoretical perspective, organisations emerge and reproduce themselves as social systems through the continued sense making, mutually related and coordinated acting of their members which itself is based on grown routines and assumed expectations.

In the course of their continuous action flow, actors may generate explicit knowledge through reflection and concept formation about certain aspects of their experiences in acting and in dealing with things as described. This knowledge can be expressed and objectified again in the form of linguistic signs, of organisational schemes or of technical artifacts. In particular, technical systems like computer artifacts can thus be designed as a product of reflection on human activities, as objectified explicit knowledge by modeling certain courses of practical action. This model formation, in principle, undergoes the following three steps of abstraction and formalisation:

Semiotisation: describing courses of action by signs as a prerequisite for communication (result: *application model*);

Formalisation: abstracting from interpretations bound to situation and context by using standard signs and operations (result: formal model, *specification*);

Algorithmisation: describing courses of action as formally computable procedures by means of the standard signs and operations (result: *computing model*).

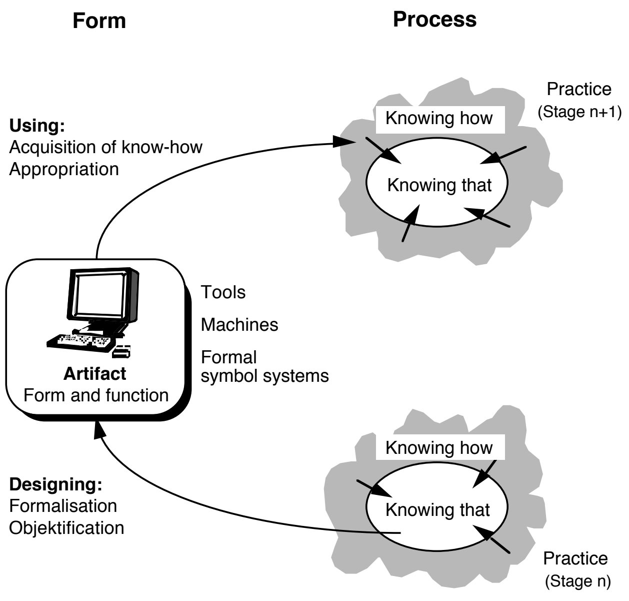

Figure 3: Genesis and use of technical artifacts: Dialectics of form and process

In this way, computer artifacts emerge as objectified propositional knowledge about purposeful human acting. They are, as such, used again as means for further acting. As "congealed knowledge" inscribed in their functions and properties, they embody aspects of human practice, and as means of work to practical ends they set specific action requirements for effective use for which they must be appropriated again. Appropriation for skillful and effective use thus constitutes a new practice, new ways of doing things ([2]; cf. Fig. 3).

Since they are derived from abstract, decontextualised knowledge, technical artifacts always contain empty "slots" that have to be filled in use through "recontextualisation", i.e. by interpretation and application suited to the situation. As a consequence, their use value is constituted in the application that is, due to the scope of interpretation within the limits of the action requirements, open for diverse use.

By routinely enacting the artifacts' forms and functions in use, they structure human action, and in this way they become involved as rules and resources in the constitution of a particular recurrent social practice. Through recurrent interaction with the artifact at hand, certain of the artifact's properties become implicated in an ongoing process of structuration in which rules and routines of using it emerge. The resulting recurrent social practice produces and reproduces a particular structure of technology use [21]. Consequently, the design and use of technical artifacts have to be regarded as integral part of social systems' dynamics and, hence, the development of organisational practices.

According to this dialectics of expressive form (objectified knowledge) and process (appropriation for use), technical acting, the interaction with computer artifacts to accomplish a given task, can be understood as a process of "social construction of reality" [8]. Since the meaning of an artifacts' functions is created through interpretation in the process of acting with them, they can also be interpreted by others acting in the same action context. Successful and mutually confirmed acting thus leads to a shared understanding among the co-workers. Like practicing a language or organisational acting, computer artifacts, thus, are socially embedded in sign processes. In all these activities conceptual knowledge is externalised or objectified as forms – be they technical artifacts, language terms or organisational schemes – together with emerging rules how to interpret and how to sensibly act with them.

The externalised forms, in turn, can be used as resources for further acting; they even enable or allow for new ways of acting, if interpreted differently. As far as the rules of acting with them are being appropriated and internalised, they establish, together with the objectified forms they refer to, a new practice. It is these mutually shared (but mostly unconscious) rules (the formative context) that enable the actors to appropriately interpret situations or facts as well as data, instruments or instructions, in short: to fluently act in the organisational environment.

The expressive forms as resources together with the rules to deal with them, i.e. the attitudes, values, ways of thinking and acting, and schemes of interpretation, constitute a social structure that enables and, at the same time, constrain collective acting ("duality of social structure"). What the actors in an organisation can imagine and which opportunities to act they see in a given situation thus depends on the expressive forms they created as well as on the interpretative rules they developed to deal with them. The actors thus are socially constructing their reality, however not of their own free will, but as prisoners of the conditions they have developed to enable and regulate their collective acting. By making sense of resources at hand through interpretation (*signification*), by sanctionising actions according to norms (*legitimation*), by influencing other actors through administrative resources or by prescribing the use of technical artifacts (*domination*), each time in these social practices they create rules that constrain the scope for future action and negotiation. The better the expressive forms are adjusted to the action context and the more appropriately they are interpreted, the more effective the social practice of collective acting can develop (2, 10, 21, 23]; see Fig. 4).

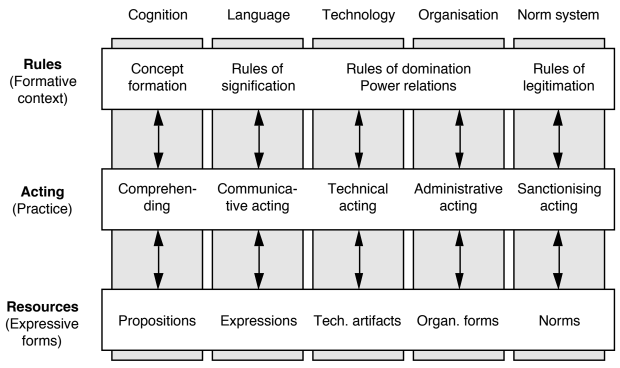

Figure 4: Structuration: Mutual constitution of acting and social structure

A paramount consequence of the semiotic nature of computer artifacts and their embeddedness in sign processes of social interaction is the indispensable fact of "double hermeneutics" [10]. In contrast to natural sciences, where (with the exception of quantum mechanics) cognition and the object of cognition are independent of each other, in social sciences observations do change their own object of observation. Hence, the object of observation, the social system, is reflexive in the sense that the explicit knowledge gained about the system – as well as the technical artifacts derived from that knowledge – becomes part of the system's resources and rules being changed by this. Social scientists, like system designers, have to interpret features of a social system as object of observation, in which they themselves take part as observers. Their thinking belongs to the same system they think about.

Formalisation and algorithmisation as central computing science activities of system analysis, modeling and design exactly are such events of observation that change the object of observation. Sign processes observed and modeled in this way, therefore, are being changed by exactly these activities: The object of modeling undergoes change by the process of modeling itself – a fact that has been almost neglected so far in software engineering with fatal consequences.

Moreover, the development of technical artifacts (and of software in particular) so far has been predominantly concentrated on processes of design according to functional requirements and almost neglected the reverse process of appropriation and enactment for effective use. However, the skill to make sense of the artifacts, to find adequate interpretations for accomplishing the working tasks is at least of equal importance and requires creative acting as well [26]. And the collective learning efforts necessary for the effective appropriation and enactment are much more expensive than design.

These are, according to the theoretical perspective presented here, the main reasons for failure in IT application projects. The following section focuses on practical consequences that can be derived from this.

CONCLUSIONS FOR IMPROVING PROJECT PRACTICES

A number of conclusions can be drawn from the semiotic perspective on computer artifacts with respect to self-comprehension of computing science as a discipline on one hand and with respect to effective improvements in the social practices of implementing and using IT systems in organisations on the other.

First, the semiotic perspective opens the mind for a new comprehension of computing science as a discipline of *technical semiotics* which allows to conceive IT systems as signal processing artifacts embedded in the sign processes of social interaction. As such they serve, provided that they are appropriately designed, adopted and enacted, as media for organising work or value creation and knowledge transformation processes. On the basis of the triadic sign concept by Peirce, it can also bridge the gap to modern sociological theories of organisations in order to gain a holistic view and integrated procedures on system design and organisational development.

Second, the semiotic perspective, thus, also delivers the key for understanding the reasons behind the permanent software crisis and the IT productivity paradox. As digital devices and media for organising, IT systems are not just models or representations of work processes but rather serve as supportive technical artifacts that, in the course of organisational development, must be co-designed, appropriated and enacted for effective use together with other organisational resources in the social practices of an organisation. Due to the self-referential nature of these activities, the social practices are themselves changed by this. Consequently, the effects produced are not solely dependent on the implemented system functionality, but are a result of how they have been socially embedded and enacted for practical use. System quality can, therefore, only be evaluated in the context of practical use.

Third, as, a consequence of this, it is indispensable to involve end users in design and implementation of both the technical and the organisational features of the new work system from the beginning. As designers normally have only little understanding of the real work tasks and procedures and users have only little knowledge about the options IT has to offer for organisational redesign, both main actors in the design and implementation process must cooperate. In order to overcome their symmetrical ignorance, they are compelled to develop a shared understanding of the underlying work processes and frame conditions. A number of practically proven methods exist to support user participation including future workshops, design scenarios or social simulation and rapid prototyping.

Fourth, as the design of IT systems is a reflexive endeavour in the sense that the systems' appropriation and use change the work processes they are designed for, frequent changes of functional requirements during system design and implementation are inevitable. Software engineering methodology, therefore, must cope with this inescapable fact and organise design and implementation processes in a reflexive or evolutionary way with iteratively revised and improved versions of the system or its modules. This requires sound methods for software engineering and project management that combine aspects of modular design, formative evaluation and collective learning with constrained range in order to confine the risks. Moreover, project management must conceive and organise the joined evolutionary design, implementation and appropriation efforts as integral part of organisational development.

Fifth, all actors involved must realise the fact that implementation and use of IT systems have strong impact on the balance of organisational flexibility and rigidity. All human acting must be sufficiently supported by routines in order to be fluent and efficient. Formal organisational procedures and routines, therefore, help to organise efficient collective acting. It actually is the purpose of organisations to reduce contingency and to confine the space of communication by rules, routines and formal procedures. And as IT systems, by definition, operate on the basis of completely determined procedures in form of algorithms, they appear as a most appropriate organisational medium. However, as they in turn impose rigid action requirements on the users working with them, they may overly constrain the necessary flexibility in action that is needed to cope with uncertainties and surprises in the organisation's environment. Hence, the actors must, during the process of integrated organisational redesign and system implementation, find a reasonable balance in this field of conflict between flexibility and rigidity.

In sum, taking these considerations together, the institution of the UsersAward and the procedures around it appear as an adequate approach to raise the consciousness for the problems presented, to intensify communication between suppliers and users and to improve the usability of systems.

REFERENCES

1. Brödner, P., 2002: Der Held von Caputh steht nicht allein. Wie Wissenschaft die Nutzungsprobleme der Informationstechnik ignoriert, in: Moldaschl, M. (Hg.): *Neue Arbeit – Neue Wissenschaft der Arbeit?* Heidelberg: Asanger 2002, 339-364

2. Brödner, P., 1997: *Der überlistete Odysseus. Über das zerrüttete Verhältnis von Menschen und Maschinen*, Berlin: edition sigma

3. Brynjolfsson, E. (1993): The Productivity Paradox of Information Technology, *CACM 36*, No. 12, 67-77

4. Brynjolfsson, E.; Hitt, L.M., 2000: Beyond Computation: Information Technology, Organizational Transformation and Business Performance, *Journal of Economic*

Perspectives 14 (4), 23-48

5. Davenport, T.H. (1998): Putting the Enterprise into the Enterprise System*, Harvard Business Review* July-August, 121-131

6. Farrell, D. (2003): The Real New Economy, *Harvard Business Review* October, 105-112

7. Floyd, C., 2002: Developing and Embedding Autooperational Form, in: Dittrich, Y.; Floyd, C.; Klischewski, R. (Eds.), 2002: *Social Thinking – Software Practice*, Cambridge (MA): MIT Press, 5-28

8. Floyd, C., 1992: Software Development as Reality Construction, in: Floyd, C.; Züllighoven, H.; Budde, R.; Keil-Slawik, R. (Eds.): *Software Development and Reality Construction*, Berlin Heidelberg New York: Springer, 86 – 100

9. Gibbs, W.W., 1994: Software: chronisch mangelhaft, *Spektrum der Wissenschaft*, Dezember 1994, 56-63

10. Giddens, A., 1984: *The Constitution of Society. Outline of the Theory of Structuration*, Cambridge: Polity Press

11. Kapor, M., 1996: A Software Design Manifesto, reprint in Winograd, T. (ed.): *Bringing Design to Software*, Reading (MA): Addison-Wesley

12. Landauer, T.K., 1995: *The Trouble with Computers. Usefulness, Usability, and Productivity*, Cambridge (MA): MIT Press

13. Maucher, I. (2001): *Komplexitätsbewältigung durcph Entwicklung und Gestaltung von Organisation*, München: Hampp

14. Maucher, I. (Hg.) (1998): *Wandel der Leitbilder zur Entwicklung und Nutzung von PPS-Systemen*, München: Hampp

15. McKinsey Global Institute (2001): Productivity in the United States, http://www.mckinsey.com/knowledge/mgi/reports/productivity.asp

16. Mead, G.H., 1903: Die Definition des Psychischen, in: Gesammelte Aufsätze, hg. von Hans Joas, Bd. 1, Frankfurt / M: Suhrkamp 1987, 83-148

17. Nake, F., 2001: Das algorithmische Zeichen, in: Bauknecht, W.; Brauer, W.; Mück, T. (Hg.): *Informatik 2001. Tagungsband der GI/OCG Jahrestagung*, 736-742

18. Nake, F.; Grabowski, S., 2001: Human-Computer Interaction Viewed as Pseudo-Communication, *Knowledge-Based Systems 14*, 441-447

19. Nardi, B.A. (Ed.), 1996: Context and Consciousness: Activity Theory and Human-Computer Interaction, Cambridge (MA): MIT Press

20. Ogden, C.K.; Richards, I.A., 1989: *The Meaning of Meaning* (reprint of 1923 edition), San Diego New York London: HBJ

21. Orlikowski, W.J., 2000: Using Technology and Constituting Structures: A Practice Lens for Studying Technology in Organizations, *Organization Science 11 (4)*, 404-428

22. Orlikowski, W.J.; Gash, D.C., 1994: Technological Frames: Making Sense of Information Technology in Organizations, *ACM Transactions on Information Systems 12*, 174-207.

23. Ortmann, G.; 1995: *Formen der Produktion. Organisation und Rekursivität*, Opladen: Westdeutscher Verlag

24. Oz, E., 1994: When Professional Standards are Lax. The CONFIRM Failure and its Lessons, *CACM 37*, No.10, 29 - 36

25. Peirce, C.S., 1903: *A Syllabus of Certain Topics of Logic*, Collected Papers, 1.180-202, 2.219-225 and other paragrafs; German translation: Phänomen und Logik der Zeichen, Frankfurt/M: Suhrkamp 1983

26. Pipek, V., 2005: *From Tailoring to Appropriation Support: Negotiating Groupware Usage*, Acta Universitatis Ouluensis A 430, Oulu: Oulu University Press

27. Royal Academy of Engineering (ed.), 2004: *The Challenges of Complex IT Projects*. The report from a working group of the Royal Academy of Engineering and the British Computer Society, London: The Royal Academy of Engineering

28. Scott, E.J.; Vessey, I., 2002: Managing Risks in Enterprise Systems Implementations, *CACM 45* April, 74-81

29. Schon, D.A., 1983: *The Reflective Practitioner: How Professionals Think in Action*, New York: Basic Books

30. Suchman, L., 1987: *Plans and Situated Actions. The Problem of Human Machine Communication*, Cambridge (MA): Cambridge University Press

31. Taxén, L.; Lilliesköld, J., 2005: Manifesting Shared Affordances in System Development – the System Anatomy, in: Ågerfalk, P.J.; Bannon, L.; Fitzgerald, B. (Eds.): Proceedings of the 3rd International Workshop on Action in Language, Organisations and Information Systems (ALOIS 2005), Limerick (Ireland), March15–16

32. The Standish Group International, 2001: Extreme CHAOS. The 2001 update to the CHAOS report, http://www.standishgroup.com/sample_research/index.php