# A Software Engineering Service Center for Scientific Software Development

Gabriele. H. Pawlitzek

*Deutsche Forschungsanstalt fr Luft- und Raumfahrt (DLR) e.V., European TeleCASE Center, Lilienthalplatz 7, 38108 Braunschweig, Germany*

Science on its way to the information society depends more and more on the utilization and thus development of leading edge software systems. In this context, software engineering expertise and technology is of increasing importance to the competitiveness of research organizations all over the world. The German Aerospace Research Establishment (DLR) has started an initiative to build up a software engineering service center for scientific software development three years ago. This paper presents the organizational and technical concept of such a service center as well as the key success factors and lessons learned during the course of the last three years. The future perspective of this service center becoming a European Facility for Applied Software Technology (EUROFAST) in the framework of the European System and Software Initiative (ESSI) is outlined and the information and service offerings available to interested organizations and companies are described.

*Key words:* Software, software engineering,, computer aided software engineering, software life cycle, software technology Introduction

## 1 Introduction

Information technology and in particular software systems provide the base for science and research on its way to the information society. This trend enables not only classical science and research domains to reach for new dimensions utilizing computer simulation, but also fosters completely new science and research domains like remote sensing.

## 1.1 *Software Engineering Trends*

The important role of software in science and research projects results in the development of large, complex software systems by researchers from almost all science and research domains. This trend leads to

- increasing software development and maintenance effort and costs as well as prolonged project schedules,
- growing problems with regard to software complexity and efficiency as well as quality and maintainability,
- researchers solving more and more computer science problems instead of scientific problems.

In addition, world wide cooperation as well as more and more cooperation with industry requires the utilization of industrial standard, established methods, and market leading tools to distributed collaboration in scientific software development. Thus, software engineering becomes one of the major cross sector technologies and in this context is of increasing importance for the competitiveness of science and research.

Based on these trends and the special circumstances and needs of a research organization, the German Aerospace Research Establishment (DLR) has started an initiative to build up a software engineering service center for scientific software development three years ago. The initiative, in the beginning called CASE Service Center, has rapidly grown over the past three years and has previously been renamed European TeleCASE Center, offering not only computer aided software engineering (CASE) services but also computer supported cooperative working (CSCW) services for scientific software development projects. In addition, the European TeleCASE Center is in charge of establishing and operating the European Facility for Applied Software Technology, which offers vendor independent expertise, services, and infrastructure in software best practice.

## 1.2 *Software Engineering Terminology*

The terminology used in this paper is based on the ANSI IEEE STD 610.12-1990, "Standard Glossary of Software Engineering Terminology", which contains the following definitions of software, software engineering, computer aided software engineering, and software life cycle:

Software

Computer programs, procedures, and possibly associated documentation and data pertaining to the operation of a computer system.

Software Engineering

The application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is the application of engineering to software.

Computer Aided Software Engineering (CASE)

The use of computers to aid in the software engineering process. May include the application of software tools to software design, requirement tracing, code production, testing, document generation, and other software engineering activities.

Software Life Cycle

The period of time that begins when a software product is conceived and ends when the software is no longer available for use. The software life cycle typically includes a concept phase, requirements phase, design phase, implementation phase, test phase, installation and checkout phase, operation and maintenance phase, and, sometimes, retirement phase.

The awareness that software is not only source code and that software development is not only software coding is vital for the establishment of a software engineering service center for scientific software development, in order to succeed in addressing the issues stated in the previous section.

## 2 Organizational Approach

The organizational approach of such an initiative is closely linked to top management commitment, which is vital to succeed in changing the attitude and workstyle in software development on a broad scale. This top management commitment includes not only the provision of the necessary start-up resources but in particular the permanent attention to software engineering issues in decision making.

However, changing software production and operation of an organization with more than 2.000 researchers developing different kinds of software systems and components requires not only top management support and excellent software engineering skills, but also change management and human interaction skills to bring together the critical mass of enthusiasm, skills, and support from all levels of management as well as from the researchers engaged in software project management, specification, design, implementation, operation and maintenance. Figure 1 shows the resulting organizations approach based on topic focused application domain working groups (ADWGs) and organization specific software process engineering groups (SPEGs), discussing what software systems to develop how. Representatives of these different groups form the DLR ADWG and SPEG, which provide the necessary link to DLR top management, to the software engineering service center as well as into the different research institutes and departments. In addition, the close co-operation of these groups in different research projects (e.g. MegaFLOW, SUPEA) is depicted.

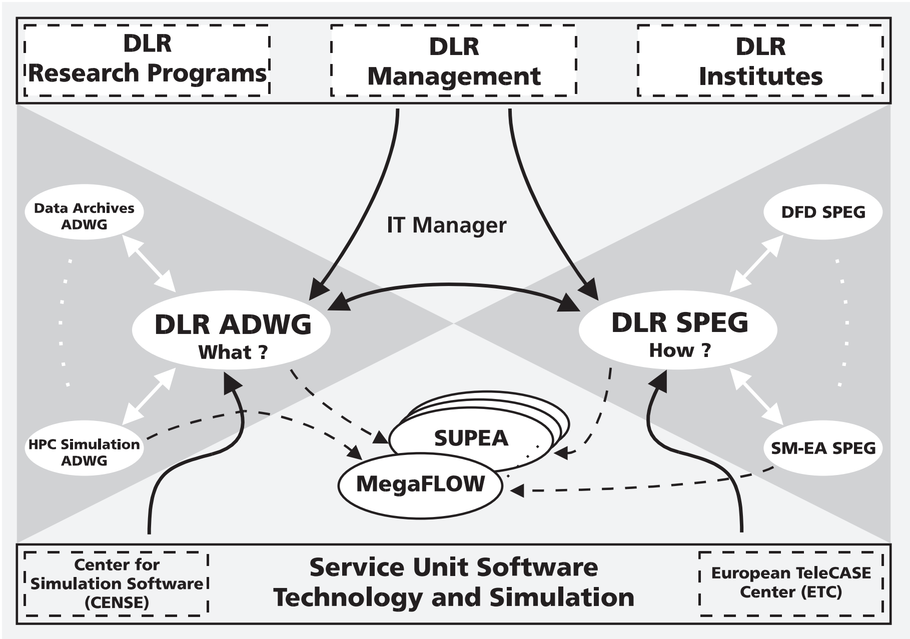


Fig. 1. Organizational Approach in Software Process Improvemen

Besides of the importance of software engineering to professional software procurement and production, it is important to take into account that leading edge research is performed by excellent scientists, which are eager to achieve their goals and to develop good quality software. A co-operative workstyle between scientists and software engineering experts is important to succeed in adapting industrial software engineering standards, methods, and tools to the specific requirements of scientific software development. This close co-operation and team spirit is vital to overcome the difficulties and drawbacks inherent to the introduction of any new workstyle and technology into an established working environment.

In this context, tailoring of project specific software development concepts is not only necessary to accomplish specific project or user requirements, but also to define a project specific software engineering approach that is achievable in terms of software development schedule and costs. The establishment of this project specific software development concept, depicted in figure 2, should be performed in close co-operation between project management, project team, as well as quality management and software engineering experts.

Experience shows, that in many cases management and team members have a too optimistic view with regard to the effort needed to introduce and adapt new standards, methods and tools. This optimistic view resulting from glossy awareness brochures and road shows from software industry often result in nicely written shelfware standards and documents as well as expensive shelfware tools. Therefore, a vendor independent consulting focusing on the essential needs of the project with regard to the necessary standards and the recommendable methods including the required training and coaching of the project team is strongly advisable. Based on the resulting quality management system and software engineering concept, an evaluation of tools and infrastructure leads to the selection of tools supporting the project goals and concepts.

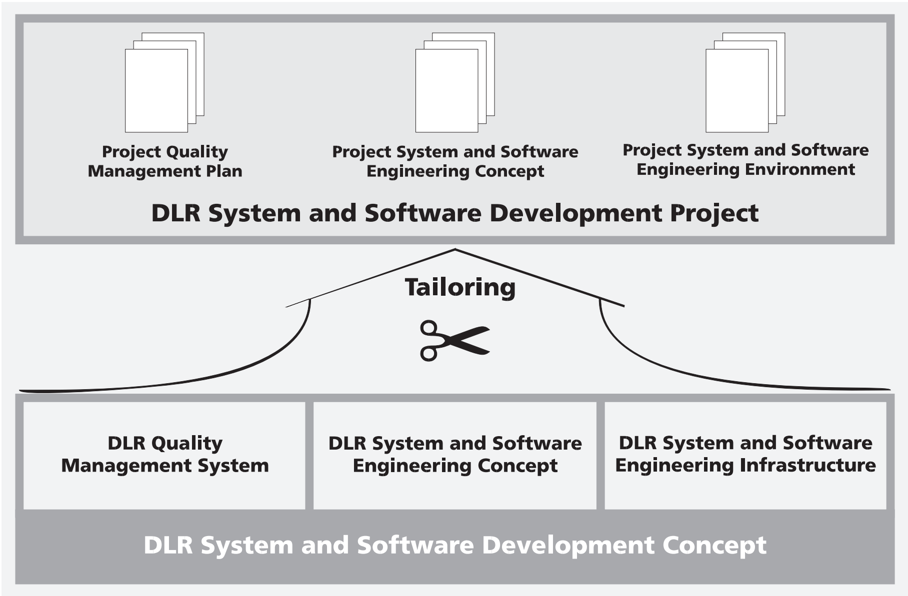


Fig. 2. Project Specific Tailoring of Software Development Concepts

However, it cannot be stressed enough that the minimum of well defined standards, carefully selected methods and thoroughly evaluated tools provides provides the best base for successful software projects. An impressive list of standards, methods, and tools introduced in a single projects often results in disappointment when looking behind the scenes of glossy project presentations, resulting in projects overrunning schedule and budget despite of quality management and software engineering activities.

## 3 Technical Approach

The technical approach towards the establishment of the software engineering service center for scientific software development is to offer expertise and practical experience in the different software engineering domains relevant to most scientific software development efforts. The resulting map of software engineering expertise is depicted in figure 3, offering software engineering services for the development of structured systems (e.g. FORTRAN, C), object-oriented systems (e.g. C++, Java), database systems (e.g. Oracle, ObjectStore) and user interface systems (e.g. OSF Motif, Tcl/Tk) during all phases of the software life cycles. This map of software engineering expertise make evident, why in order to deliver a professional software engineering service for scientific software development a critical mass of about 12 team members is needed to build up the necessary in-depth practical expertise. In addition, the rapid progress in software technology requires the software engineering service team to permanently acquire leading edge knowledge well in advance before asked for support by the project teams.

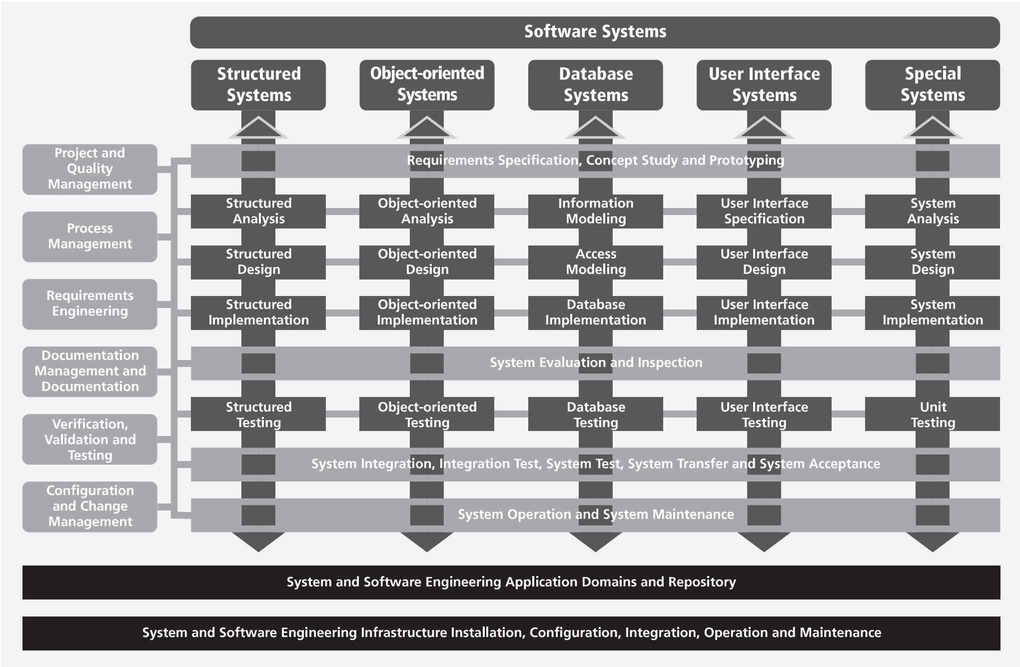


Fig. 3. Map of Software Engineering Expertise

The technical approach further includes the set-up and provision of a software engineering tool pool, depicted in figure 4, offering tools from the market leading vendors for evaluation and application by the different projects. This includes accompanying services like tool consulting, coaching, and training as well as a first-line helpdesk support. Together with ready-to-use tool configuration and integration solutions as well as service offerings to configure and integrate project specific software development environments, the software engineering service center enables project management and teams to focuse on the scientifically challenging research domain problems and acquire expert support in software engineering for scientific software development.

However, this technical approach and services have to be based on the organizational approach described in the previous section. Lack of management commitment or change management skill are crucial and will almost in any case lead to unsatisfactory results with regard to solving the problems addressed in the introduction of this experience report.

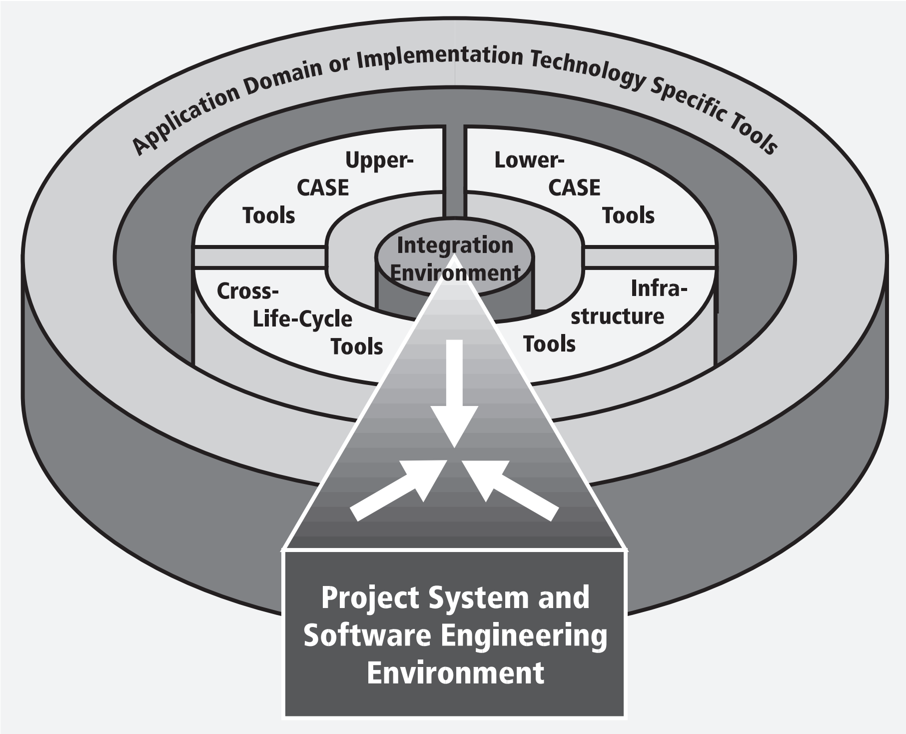


Fig. 4. Software Engineering Tool Pool

During the course of the last three years based on the requirements of the customers, i.e. the scientific community served by the software engineering service center, several service offering have been established to better serve the need of scientific software development projects. In order to provide an impression, which kind of services are offered and which advantages are involved with the utilization of these services, the following list contains brief descriptions of some service offering examples:

- analysis and design of software systems

  The iterative analysis of the problem domain followed by the design of the software system provides the base for implementing maintainable software systems. The services offered to support computer aided software engineering based on state-of-the-art software engineering methods comprise consulting, coaching, and training in software best practice methods and tools. In addition, more and more projects require full support based on team members from the software engineering service center joining the project team and performing the software engineering work packages and tasks during the course of the project. The resulting automatic documentation of system analysis and design models is depicted in figure 5.

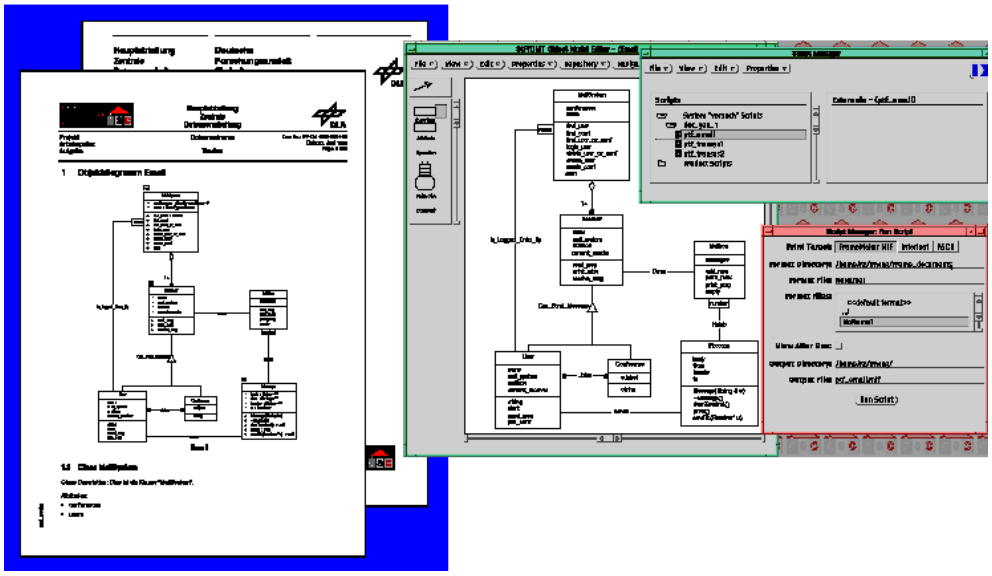

Fig. 5. Object-oriented analysis and design

- specification and realization of software components

  The intensive co-operation between the scientific community and the software engineering service team resulted in a service offering not even envisaged at the start-up of this initiative. Today, the ready-to-use specification and realization of software components requiring intimate software engineering expertise and practical experience engages already about one third of the software engineering service team members. This software components are usually subsystems of large scientific software systems and focus on areas like the graphical user interface, the communication middleware or the embedded database.
  The professional in-house development of these software components solves the legal issue with regard to the technology transfer of software systems as well as the maintenance problems by providing well defined and documented software components for the integration into large scientific software systems.

- static and dynamic analysis of software source code

  The application of static and dynamic source code analysis supports significant improvements in software quality by detecting more than 50% of problems and bugs inherent in a software source code. This results in significant reductions in debugging and test effort and over the time improves coding practice of the project team by providing permanent feedback on the code quality achieved. In addition, the information gathered by these tools provides an excellent base for performing reengineering activities on a broad scale. The outputs generated by different static and dynamic analysis tools are shown in figures 6 and 7.

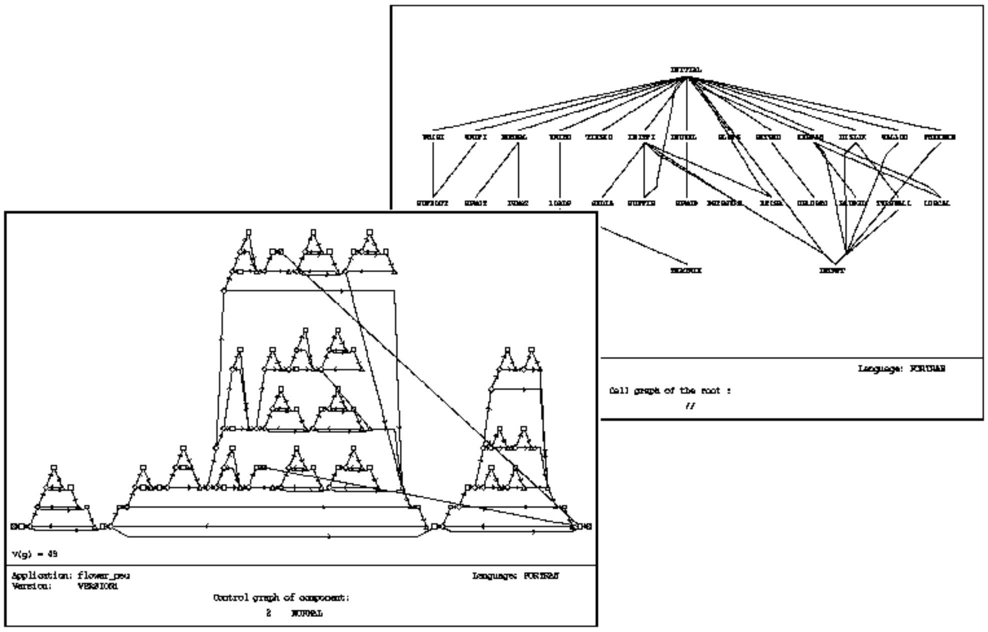

Fig. 6. Static analysis of the source code structure

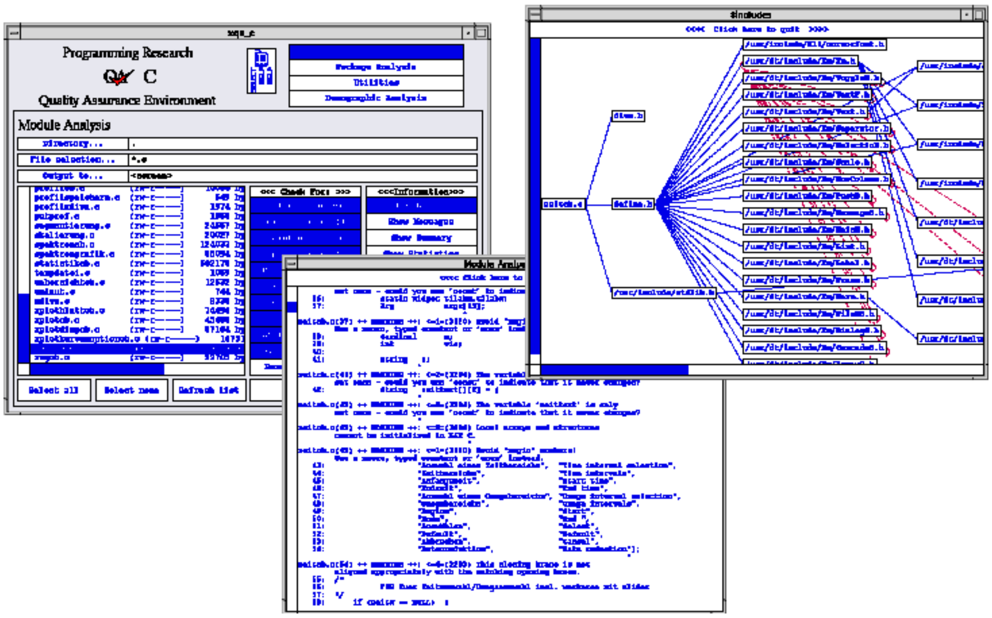


Fig. 7. Static analysis of the source code quality

## 4 Projects

The practical experience gained in a variety of scientific software development and maintenance projects resulted in more and more projects initiated to improve the state-of-the-art provided by the software engineering industry in order to better support the specification, implementation, and testing of large and complex, technical software systems in science and research. The most important initiatives and projects funded by the European Commission are described briefly in the following sections.

### *4.1 European Facility for Applied Software Technology*

Bringing together practice and theory in developing software is the crucial point in software engineering. For this purpose, an European Project has been started to provide assistance for practitioners in software engineering. EUROFAST, the European Facility for Applied Software Technology depicted in figure 8, will be a platform for information exchange between users, experts, and tool providers. EUROFAST offers access to real- world case studies and examples for practitioners bringing together expertise, methods and technology that provide results in real world projects. EUROFAST provides access to material generated by these real world projects. This material does not only consist of the code, but all the products generated during the software life cycle, e. g. models of the problem analysis and system architecture, test suites and of course documentation samples such as requirements specifications, design descriptions, test procedures and reports, as well as project plans. Starting from these best practise examples, EUROFAST offers the infrastructure and assistance to evaluate methods and tools to meet the needs of the user. An international panel of experts will be available to provide advice.

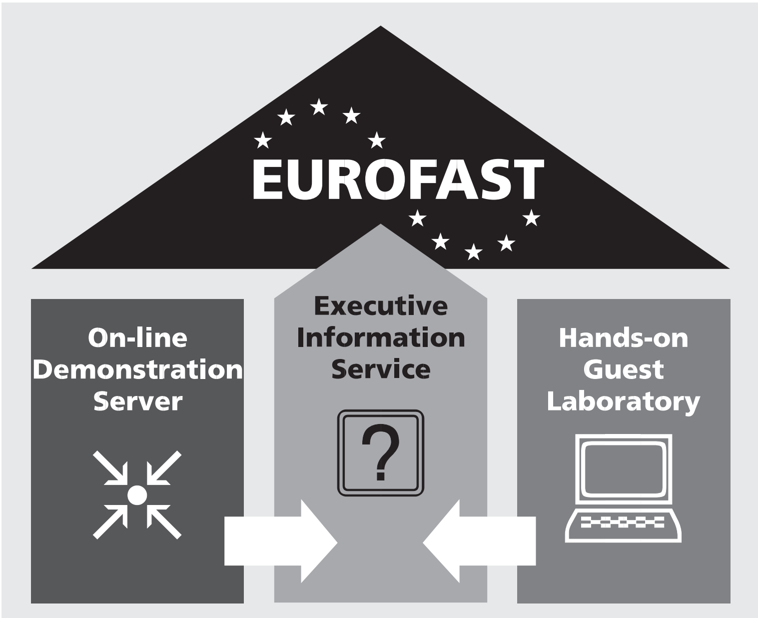


Fig. 8. European Facility for Applied Software Technology (EUROFAST)

EUROFAST provides information proved in practise, from experts, about these topics. These topics are not only viewed in isolation, assessment is made of their capabilities for working together. In software development projects, one of the first tasks is to choose the most applicable methodology to be followed for the problem domain. Then, in most cases, there are advantages in using a tool which supports the methodology chosen, because today's software projects are too complex to be managed manually. The evaluation material being published by EUROFAST will be prepared and released by vendor-independent experts to guarantee vendor neutrality in the information provided.

EUROFAST will assist this definition, selection, and evaluation process by providing information about

- techniques and methods, and their adaptability to different software problem domains,
- technology and tools, and their scope and usability for the application in real world projects, and
- how to adopt and adapt methods and tools to profitably support specific problem domains.

EUROFAST will provide the following service to effectively and efficiently support these activities:

- Technology Transfer Node
  The technology transfer node provides consulting and service with regard to the application for European and national project funding when adopting information technology or software best practice.
- Executive Information Service
  The executive information service provides off-line and on-line information on software best practice as well as contacts to other experts and users via different events and mechanisms.
- On-line Demonstration Server
  The on-line demonstration server provides on-line demonstrations on the utilization of computer aided software engineering based on real world examples and case studies.
- Hands-on Guest Laboratory
  The hands-on guest laboratory enables users to evaluate market leading tools based on the existing real world examples and case studies, but also according to their individual requirements and needs.
- Research and Development Laboratory
  The research and development laboratory provides customer specific leading edge solutions in software technology based on the experience gained during the course of a large number of real world projects from different application domains.

EUROFAST will establish a European network of vendor independent organizations bringing together users, practitioners, providers and researchers, providing them with access to the EUROFAST materials and services. The network will consist of National Access Points (NAPs) providing local language assistance in their country by supplying information material, contacting experts and vendors as well as organizing local events.

### *4.2 Reengineering for Technology Transfer (RETECTRA)*

The German aerospace research establishment (DLR) coordinates large high technology software development projects and is responsible for the integration of highly sophisticated, innovative software systems and their technology transfer into an industrial environment. In this context, it is the task of DLR to integrate third party prototype software into large operational software systems. While a forward engineering process based on the software engineering standards of the European Space Agency (ESA) and the V-model is defined and implemented, the reverse engineering and integration activities necessary to integrate externally developed prototype software are neither documented nor defined. However, during the last years the increasing cooperation in large research projects produced a shift in effort at DLR, where more and more integration activities replace in-house development activities. Thus the definition and implementation of a state-of-the-art strategy and process for reengineering and integration of software prototypes developed by industrial research departments, small and medium size enterprises, research organizations, and universities is vital for the future competitiveness of DLR as integrator of large, complex and innovative software systems fulfilling industrial quality requirements.

Since these problems are common to the distributed development of large scientific software systems, this project is being funded by the European Commission in the framework of the European System and Software Initiative (ESSI) as a so-called process improvement experiment.

This reengineering process, based on state-of-the-art process and product metrics and supported by an underlying reengineering environment, shall increase the efficiency of software prototype evaluation, redocumentation, redesign, recoding and regression testing, and thus improve the productivity and competitiveness of DLR as an integrator of large, innovative software systems. Up to now, the absence of a defined reengineering and integration process based on established software engineering standards and methods results in quite different evaluation results and integration efforts, depending on the individual team member performing the different software engineering activities. In addition, the products and documents achieved by these reengineering activities are not compatible with the corresponding results from the forward engineering activities. This facts result in error prone software systems that are difficult to maintain and that over the time consist of more and more subsystems that are documented differently. Therefore, it is important, that the reengineering process to be defined and realized in the framework of this process improvement experiment is closely integrated with the existing forward engineering process depicted on the page before.

Figure 9 shows the required re-engineering process based on established software engineering practices, standards and metrics. In this context, the training and coaching of the team members having to apply this process and the underlying procedures is essential in order to achieve a more efficient and effective reengineering process in terms of resulting quality and necessary effort. Finally, in order to achieve a practicable process it is necessary to provide tool support for those tasks of the reengineering process, that are to time consuming if performed manually. These tasks include the analysis of the program and data structure of the software prototype, the generation of product metrics, as far as possible the redocumentation of the software prototypes, and last but not least the verification and validation of the resulting software packages. The results will be published in several guides to help other organizations to adopt the approach.

## 5 Conclusion

The success of the European TeleCASE Center and its software engineering service center for scientific software development results from the following key success factors, which together from the base for a center of excellence in applied software technology for science and research:

- top management commitment in terms of decision making and funding
- focus on change management and human interaction issues co-operative, team-oriented approach based on a customer and service supplier relationship
- establishment of expertise and services based on customer requirements

This approach takes into account that software is mainly peopleware and brainware. Thus, it is of importance to convince the scientists that software engineering applied professionally helps them in achieving their research goals. However, based on the software technology currently available, this requires remarkable effort in order to achieve what the glossy awareness brochures of the tool industry promises. Therefore, a realistic, vendor independent view on the pro's and con's of software engineering standards, methods, and tools is of vital importance in order to achieve the goals of improving software quality or reducing software development and maintenance costs.

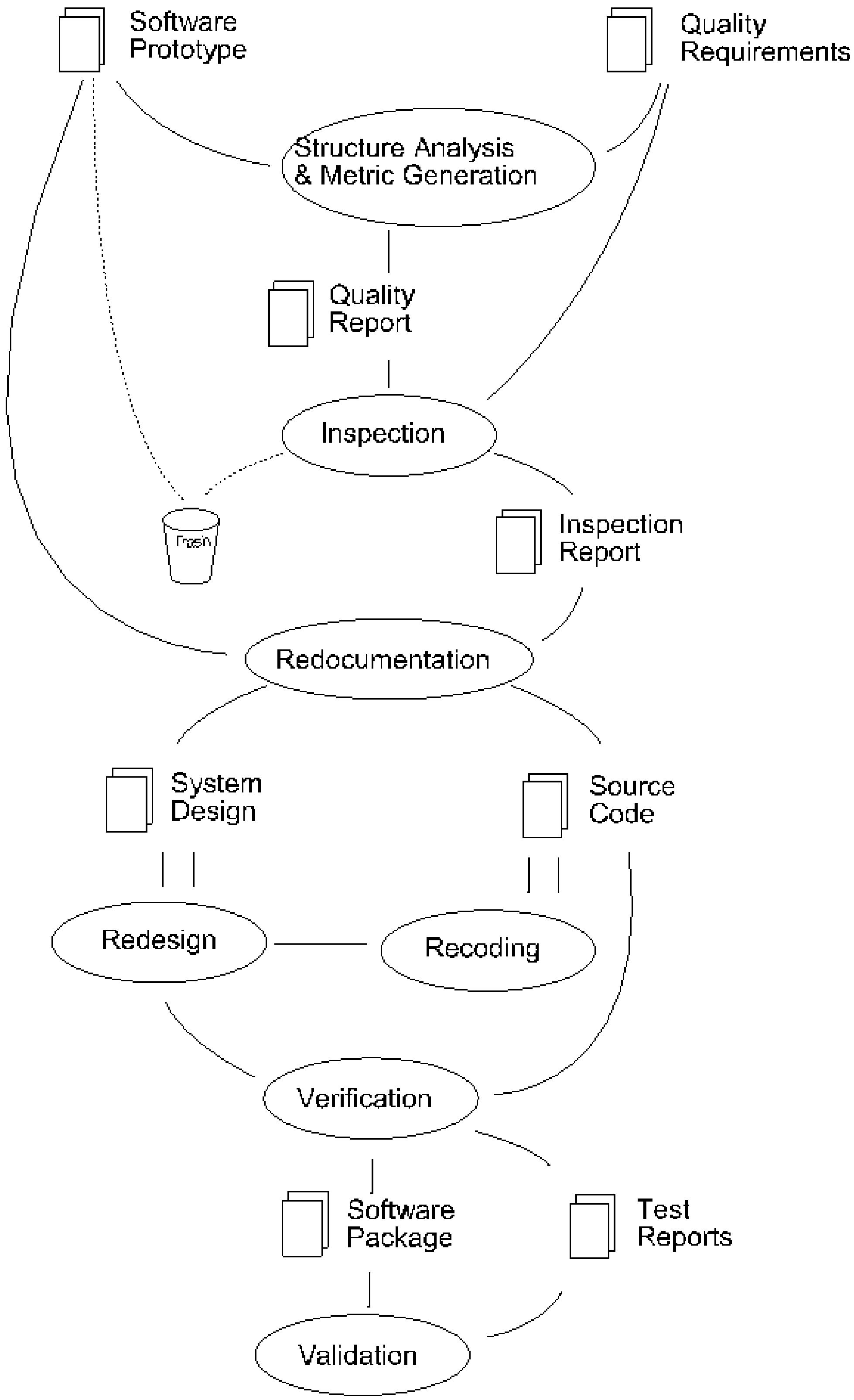


Fig. 9. Software Evaluation, Reengineering, and Integration for Technology Transfer