Zeitschrift: Comtec : Informations- und Telekommunikationstechnologie = information and telecommunication technology

Herausgeber: Swisscom

Band: 76 (1998)

Heft: 5

Artikel: NETPLAN : a framework for the development of network planning, design and optimization applications

Autor: Messmer, Bruno T.

DOI: https://doi.org/10.5169/seals-877301

From the Exploration-Programmes of Corporate Technology (8)

NETPLAN: A Framework for the Development of Network Planning, Design and Optimization Applications

In today's competitive environment, a successful network operator must offer networks that meet the quality requirements of the customer at minimal costs. Planning and optimizing are two important processes in the attempt to build new and cost-efficient networks. Although there exist numerous commercial tools for network planning and optimization, they are often non-extendible and built for a restricted set of network architectures and problems. For this purpose, we built a flexible, object-oriented software framework for the development of integrated network planning and design tools.

ATM and IP based networks support various applications such as audiovisual and data applications with very different traffic patterns. To make best use of network resources and at the same time guarantee the requested network quality, the Exploration Programme* "EP97-8 Traffic Management" investigates traffic control mechanisms, real time traffic management, routing, network dimensioning and network optimization for these networks.

* Exploration Programmes are commissioned by the Swisscom Board of Directors and executed in Corporate Technology. The activities have a mean to long term time horizon (2–7 years) depending on the area.

BRUNO T. MESSMER, BERN

The framework NETPLAN implements several reusable components that are essential to network planning and design, such as a generic network model, graphical user interfaces, algorithm and controller components. It also provides standard interactions between these components, hence allowing an application programmer to integrate new applications into a framework application easily.

Introduction
With the upcoming worldwide liberalization of the telecommunication market on the one hand and the increasing demand for a variety of communication services on the other hand, planning, design and management of telecommunication networks has become a vital issue for network operators and service providers. Telecommunication networks must be designed considering both installation costs and required capacity. Although there exist numerous commercial tools for these tasks, they all have one major disadvantage. To an application programmer, they present themselves as a black box. While they offer a certain range of solutions to problems such as backbone design, clocking analysis, etc., they cannot be adapted to integrate new optimization algorithms or special analysis methods. Hence, an open, object-oriented system for the development of planning, design and optimization applications is needed.

One possibility to build such a system is to create a set of program libraries containing various planning, design and optimization methods, similar the well-known mathematical libraries in FORTRAN. New applications are then built by calling functions contained in the program library. However, this approach limits reuse of program code to single function calls. The interaction between objects in the program must be designed for each application from scratch, hence, making the integration of applications in a standard environment very difficult. This, however, is crucial in a problem domain where individual applications share a large number of standard features. For example, most applications dealing with networks require the possibility to query, display, and modify the network configuration data. Using only a program library, the application programmer would have to rewrite the basic editing features of the application every time. Frameworks are an elegant and increasingly popular way of solving these problems [6, 3] (see Box "Software Frameworks").

In this paper, we present the framework NETPLAN [7] and a number of applications that have successfully been built on top of it. NETPLAN is based on object-oriented technology and extensively uses the paradigm of design patterns [1]. So far, we have made the experience that NETPLAN facilitates the development of new, tailor-made network applications and, in particular, allows the integration of these new applications, resulting in a high degree of acceptance by the application users. Based on NETPLAN, prototypes for different network planning tools have been developed successfully.

The NETPLAN Framework
Whether the concept of a framework is applicable to a certain problem domain strongly dependents on the structure and the shape of the applications that come from this domain. That is, if each application in a domain would require completely different data structures and unique processing steps, it would make little sense to build a framework for this domain, as only few common structures can be reused. The domain of network planning and optimization, however, is a well structured area, in which most applications share a large number of common features. For example, most applications in this domain need to build a network model, save and retrieve net-

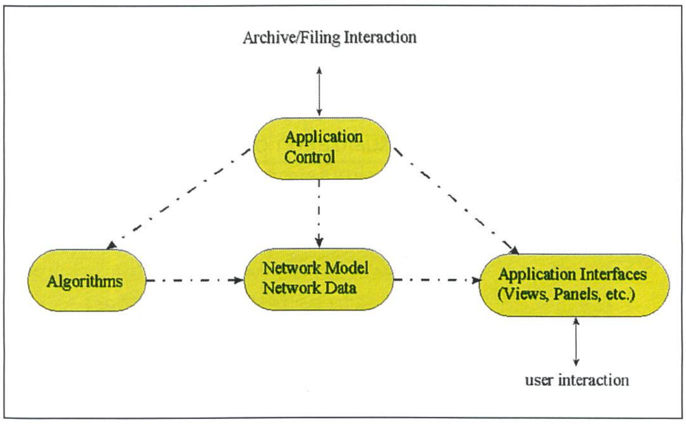


Fig. 1. Basic components of a network planning and optimization application.

work data, visualize the network components, provide the interactions between the model and the graphical user interfaces, and so on. It was one of the major objectives of the project to build a framework which incorporates all of the above mentioned features as well as some additional sophisticated components that help in building customized network applications in a fast, reusable and extendible manner.

Basically, there are four components in every network planning or optimization application, namely an application control, a set of algorithms, the network model and the user interfaces such as a network editor. In Figure 1, the four components and their mutual interactions are displayed.

The NETPLAN framework provides for each of these components a corresponding framework structure, which already implements a number of standard features. Furthermore, the interactions between the components are also provided for by the framework. For example, NETPLAN contains a generic network model, which supports the concepts of nodes, links, demands, connections, subnetworks and network interfaces. Network aspects which are specific to an application can then be added by associating network data to these predefined network objects. Also, the interactions between the network and the graphical user interfaces (operations such as move, select, zoom, add and delete) are already implemented. In Figure 2, the framework components along with the embedded user application components described in Figure 1 are shown.

It is important to note that unlike a library of function calls, the framework itself represents a complete application, the so-called standard framework application. This standard application contains various features such as a version tree editor, an advanced network editor, archiving functionality for the created networks, basic traffic routing and route optimization schemes and an algorithm management facility. User-defined application with specific behavior such as the ones described later in section 3, can then be built by either inheriting from framework components or by registering new application components with the framework (see also Box "Software Frameworks").

The most important advantage that results from using the framework, is that user-defined applications are created as add-ons to the standard framework application. Hence, all of the above mentioned features are automatically available in the user-defined application. As a consequence, the time spent in developing new applications can be reduced substantially. Furthermore, each component of a new application that is plugged into the framework conforms to the general framework rules of inheritance and composition. Therefore, modules that are created for one application may easily be reused for another application. For example, if a new algorithm is implemented that computes the delay of IP packets for an application that analyses HTTP delay behavior, then that same IP delay algorithm may later be reused in a new application that studies Internet telephony. In general, the reuse of components is strongly encouraged by the framework and helps to reduce redundancies in development projects.

Clearly, all of the above mentioned characteristics of NETPLAN would be of minor interest if the framework would not be open, fast, extendible and also portable onto different operating system platforms. For this reason, NETPLAN was realized and implemented using C++ and the commercial graphical user interface library ILOG Views. Thanks to ILOG Views, it was possible to create a framework that is both completely portable between Unix and Windows 95/NT platforms while at the same time it offers sophisticated graphical user interfaces. In order to develop new applications based on NETPLAN, the following prerequisites are required: the NETPLAN C++ Source code, a C++ compiler (MSVC++ 5.0 or GNU CC), an ILOG Views 2.3 along with the ILOG Views libraries and some experience with C++ application development.

For the purpose of demonstrating the applicability and usefulness of NETPLAN, some sample applications are briefly described in the following section.

NETPLAN in Action: Sample Applications

In order to see the diversity of applications that can be written on the basis of NETPLAN, we present three different prototypes that were created during the past 2 years by a number of different NETPLAN users. For each of these prototypes, we briefly describe the purpose, the basic structure and the functionality of the prototype application.

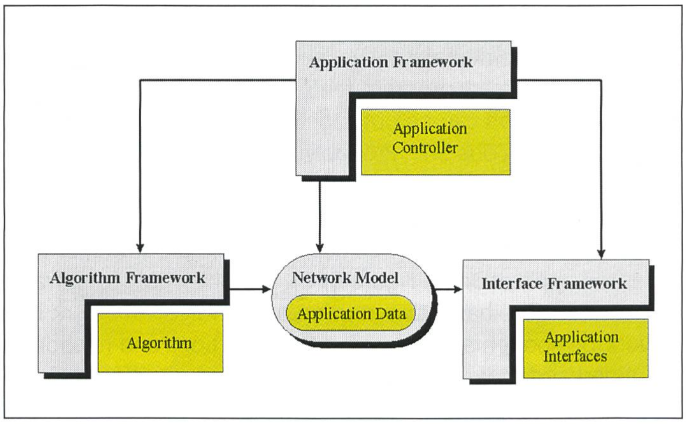


Fig. 2. Overview of the framework structure.

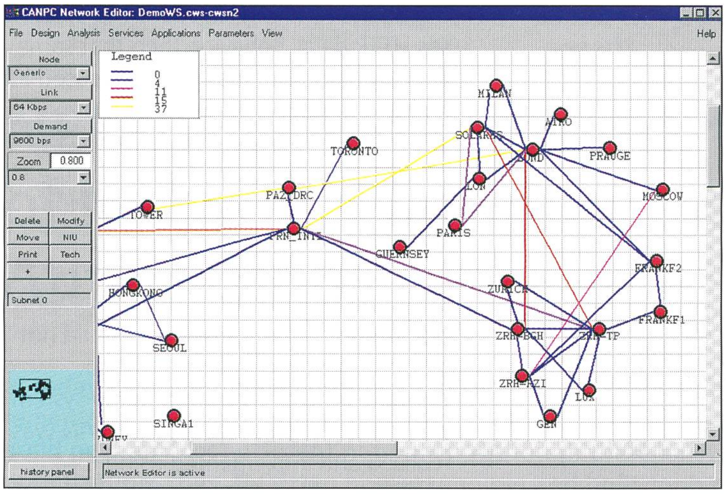


Fig. 3. Visualization of the results of a link failure simulation run.

Software Frameworks

The standard method of writing computer programs is to create for each aspect of the intended application a piece of program code, along with a main controlling body which handles the initialization and the finalization of a program run. Most commonly, the program is also enhanced by using function calls to specialized libraries such as mathematical or statistical libraries. Such a program is called a purely procedural program. In Figure 6. A), the procedural programs are illustrated with the main application body on top of a layer of libraries. Clearly, while this scheme is feasible for small programs and prototypes with little complexity, it is not advisable to build large applications with complex requirements such as graphical document handling, complex interactions between different components and nested data structures in this manner. A first step towards simplifying application development has been provided in Windows-based environments, where most programming languages provide a standard event-loop structure that takes care of the handling of the user and windows events in the system. In these event-driven programs, illustrated in Figure 6. B), the application code has been relieved of dealing with event queues and event call sequences. Frameworks, however, go one step further. A software framework, regardless of the domain it was built for, provides template components for all basically necessary program features. Hence, with a software framework, a standard application can be put together without the need for the programmer to write a single code line. Probably the most well known software framework is the set of Microsoft Foundation Classes (MFC). In an MFC application, all the necessary components that make up a document-based windows program are already present and can be adapted by the programmer to suit the needs of the desired application. In Figure 6. C), the philosophy of a framework application is illustrated. The basic idea in using a software framework is to reuse the framework components along with the interactions and relationships that are already defined and to write only the specialized application code.

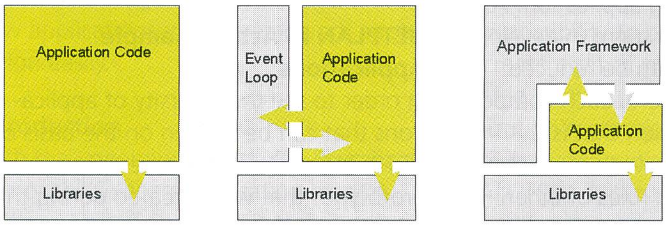


Fig. 6 A Procedural Programs

Fig. 6 B Event-driven Programs

Fig. 6 C Framework Applications

Visualization of Single-Link Failure Analysis

One of the first applications that was implemented on the basis of NETPLAN, was the visualization of the results from a link failure simulation that was performed by a commercial network planning tool. At that time, an external customer of Swisscom wanted his world-wide leased-line network to be tested with respect to reliability and survivability. The commercial tool that was used for this purpose simulated the failure of each link and analyzed for each traffic demand whether it could be rerouted in the remaining network. As a result, a list of links was presented, along with the number of demands that could not be rerouted if the listed link would fail. Due to the fact, that this commercial tool was not open for modifications, there was no possibility to graphically display the results of the simulation. Hence, in order to better serve the need of the customer, it was decided to visualize the results using the facilities of NETPLAN.

For this purpose, it was necessary to add the following components to the framework: a file interface, which enabled the application to read the network format of the commercial tool; an application panel for displaying the list of links in a tabular manner; a coloring scheme for the links, which colored each link according to the number of traffic demands that were lost when the link failed. In Figure 3, a screen shot of the visualization application is given.

Due to the fact that the network model, the network editor and the link coloring facility were already provided by NETPLAN, it was possible to save more than half of the development time that would have been necessary if the application would have had to be written from scratch. In the case of this application, the fast development time was particularly important because the external customer was eager to get a hold of a visualization of the link failure analysis results as quick as possible.

Routing Optimization

During the studies that were performed for the application described in the previous section, the problem of routing constant bit rate (CBR) traffic demands was encountered. Clearly, from the point of view of a network operator, it is important to route traffic demands in such a way that a maximum number of de-

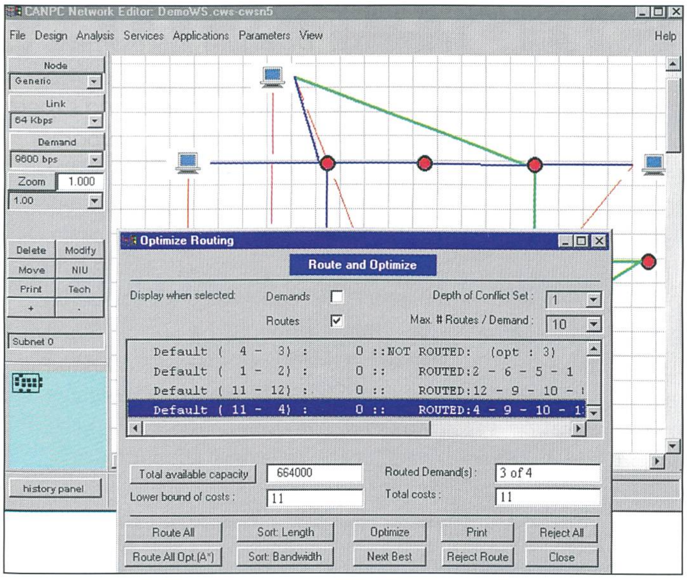


Fig. 4. Routing application panel and corresponding highlighted route (in the network editor).

mands can be satisfied. This is best ensured if the goal in routing is to reduce the usage of bandwidth in the network. However, although the problem of optimally routing a set of traffic demands in a network is easily described, it is difficult to find solutions for this problem for large networks. In fact, this problem requires in the worst case a number of computation steps that grows exponentially with the size of the network. Hence, in practice, due to limited computation time it is close to impossible to compute an optimal routing configuration for a set of traffic demands. Most routing methods in today's networks therefore lead to non-optimal and, possibly, very costly solution. Assuming that, at some point in time, it is possible to recompute certain routing decisions and initiate rerouting of certain demands, it was decided to develop a heuristic algorithm which starts from an non-optimal routing configuration and tries to gradually improve the configuration. The algorithm was designed and developed together with the University of Bern.
Due to the fact that a network model, a network editor and a basic shortest path routing scheme was already implemented in the framework, it was possible to fully concentrate on the intrinsics of the new optimization algorithm. In Figure 4, the application panel that was explicitly designed for the routing optimization application is shown (partially hiding the regular network editor displaying a simple network). Notice that the panel offers such functionalities as routing all demands, optimizing a configuration, or displaying the total available bandwidth in the network for the current configuration.
Because all of the components in the new application were inherited from existing framework classes, it was decided to make this optimization application part of the standard framework application. Hence, the new features can also be applied and reused by other, yet to be written, applications that are based on the framework.

Resource Reservation

In the past, renting constant bit rate services has been the main possibility for companies to satisfy their internal traffic requirements. With the increased promotion of ATM services, however, the interest in services that match the complex traffic patterns of companies better than a simple constant bit rate solution, has been growing steadily. One such service, for example is the temporary constant bit rate service (TCBR). With this service, the customer can specify a traffic profile, which consists of a set of traffic activations. Each such activation is described by a start and an end time and a required bit rate that remains constant over the specified time span. The activations may be subject to a repetition mode that is either daily, weekly or monthly. Clearly, with such TCBR traffic descriptions, it is possible for a customer to tailor its telecommunication contracts much better according to its actual needs than with a simple CBR service.
For the operator, on the other hand, the routing and allocation of these TCBR requests and the management of the available bandwidth profile is far from being a trivial task. For example, the routing of a TCBR request requires during the search for a route that the activations in the TCBR request are compared with the available bandwidth of the links based on a time schedule. That is, for each link, a profile of accumulated TCBR profiles must be maintained, such that new TCBR requests can be fitted into the remaining pattern of yet unallocated bandwidth of a link. It is easy to see that

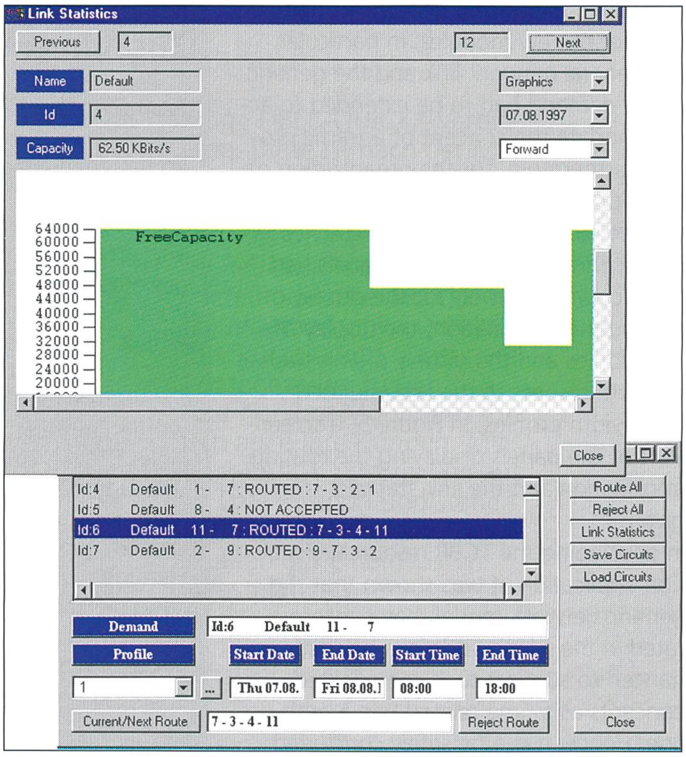


Fig. 5. Application panels for the display of link statistics and routing information.

Bibliography

1. E. Gamma, R. Helm, R. Johnson, J. Vlissides. Design Patterns – Elements of Reusable Object-Oriented Software, Addison-Wesley, 1995.
2. P. Gautier, T. Gruber. Generating Explanations of Device Behavior using compositional modeling and causal ordering, to be published, 1996.
3. G. Booch. Designing an application framework. Dr. Dobb's Journal 19, no. 2, 1994.
4. ILOG Views – Reference Manual Version 2.3. 1996, France.
5. B. Falkenhainer, K. Forbus. Compositional Modeling: finding the right model for the job. Artificial Intelligence 51, 1991, pages 95–143, Elsevier.
6. The power of frameworks. The Taligent Reference Library; Addison-Wesley, 1995.
7. B. Messmer. CANPC. An object oriented framework for network planning and design. Software Documentation, CIT-Swisscom.

without the help of a sophisticated software tool, the allocation of TCBR requests in a medium size network becomes an impossible task.
In order to gain experience, a prototype was built on the basis of NETPLAN. While it was possible to reuse components such as the basic network model, the network editor and the shortest path algorithm, it was also necessary to add a number of new modules. In particular, the generic network link and the generic traffic demand had to be extended to accommodate the concept of a traffic profile. Furthermore, for the organization of these profiles, a new controller was needed along with an archiving scheme. Finally, three new graphical panels had to be created, namely for the display of the routing information, the display of the profiles and the display of link statistics with respect to the accumulated bandwidth profiles. In Figure 5, a screenshot of the panels displaying the routing information and the link statistics is given.
In developing the TCBR resource allocation application, it was shown that the generic network model that is incorporated in the framework, can be easily enhanced to support such complicated concepts as a time related bandwidth usage. Hence, it can be safely concluded that the model is general enough to support a variety of future network extension.

Conclusions

Based on the experiences from the previously mentioned projects, it can be concluded that there are three major advantages in using the framework:

- The framework is well suited for the fast development of prototypes which can then be used to further develop a productive version. Hence, the time lost on coding that is often experienced when using languages such as Visual Basic for prototyping does not occur when a framework like NETPLAN is applied.
- It is easy to integrate different applications within a master application. Hence, a user may analyze, optimize or merely study the same network in conjunction with applications that were developed by different parties without having to switch applications and keep copies of the network in each of these applications.
- The facilities provided by the ILOG Views library make the framework truly portable. This is particularly important for applications that must run both in a backoffice environment such as a Unix system and also on the customer front, where Window-based platforms are more popular.

Dr. Bruno T. Messmer *is an artificial intelligence and software engineering expert working for the Corporate Technology Unit of Swisscom. Before joining Swisscom in 1996, he received a doctoral degree from the University of Berne for his work in the area of pattern recognition and graph matching. He has an ongoing interest in object-oriented technologies, Java and C++, Internet applications and, in general, the application of AI techniques in the telecommunication domain.*

The benefit that results from using NETPLAN has also been recognized outside of Swisscom. Consequently, the commercialization of the framework is currently under investigation. Furthermore, a demo version of the NETPLAN standard application will be integrated on the demo CD-ROM of ILOG as a successful project in the Telecom area. The demo CD-ROM from ILOG is distributed worldwide, hence, wide reaching positive publicity is guaranteed. For more information on NETPLAN, contact the author or visit the project homepage http://canpc.vptt.ch. 9.4

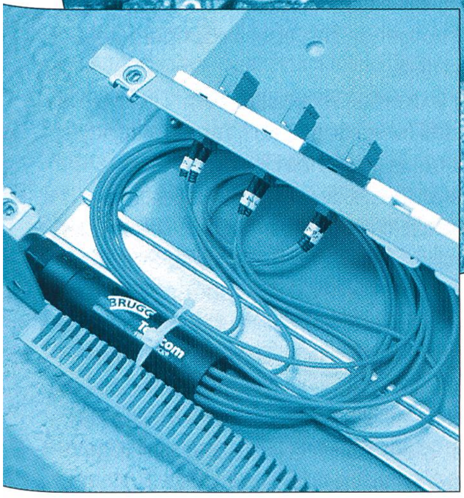
Zusammenfassung

Framework NETPLAN

Im heutigen liberalisierten und hart umkämpften Telekommunikationsmarkt muss ein erfolgreicher Netzbetreiber Netzwerke und Dienste betreiben und offerieren, die einerseits die Qualitätsansprüche der Kunden befriedigen, anderseits aber auch kosteneffizient sind. Die Planung und die Optimierung von Netzen sind in diesem Zusammenhang zwei unerlässliche Werkzeuge. Obwohl es verschiedene kommerzielle Softwarepakete für die Planung und Optimierung von Netzwerken gibt, ist festzustellen, dass die meisten dieser Programme nur für klar definierte, beschränkte Problemstellung oder für proprietäre Netzwerkarchitekturen geschaffen wurden und sie keine offenen Schnittstellen, die Erweiterungen ermöglichen würden, unterstützen. Aus diesem Grunde haben wir ein Objekt-orientiertes, flexibles Software Framework entwickelt, welches dazu dient, Netzwerk Planungs- und Optimierungsapplikationen zu implementieren. Das Framework NETPLAN stellt verschiedene wiederverwendbare Komponenten, wie einen Netzwerk-Editor und ein generisches Netzmodell, zur Verfügung, die wichtige Bestandteile jeder Planungs- und Optimierungsapplikation sind. Dank dieser vordefinierten Module kann die Entwicklungszeit für neue Softwareapplikationen im Netzplanungsbereich erheblich reduziert werden. Ausserdem ist es möglich, neue Komponenten auch für andere Applikationen wiederzuverwenden, da das Framework die Interaktionen zwischen den Komponenten vordefiniert und standardisiert.

Spitzenleistungen in der Übertragungstechnik

«Auf»Schalten zur Zielfahrt



Wer in der Formel 1 schon beim Training schnell ist, startet aus der Pole Position. Immer kürzere Zeiten werden auch beim Bau und Betrieb von Kommunikationsnetzen gefordert. Früher standen bei der Installation und Messung von Glasfaserkabeln nicht der Faktor Zeit als vielmehr Spezialkenntnisse und teure Geräte zur Diskussion. Mit FIBER-QUICK® liefern wir Ihnen Kabelverbindungen mit bis zu 48 Glasfasern und fixfertig montierten Steckern an. Sie bestellen einfach die Kabellänge mit der gewünschten Steckerzahl und erhalten von uns eine fertige FIBER-QUICK®-Verbindung. Diese ist nach der Montage sofort betriebsbereit. Zeitaufwendige Spleissarbeiten und Messungen entfallen. Mit FIBER-QUICK® schicken wir Ihnen modernste Technik anschlussfertig franko Domizil. Über kürzere und günstigere Montagezeiten freuen sich nicht nur Ihre Monteure, sondern auch Ihre Kunden. Mit FIBER-QUICK® starten Sie aus der Pole Position und stehen schon kurz nach dem «Auf»-Schalten auf einem guten Podestplatz.

BRUGG
Telecom

Brugg Telecom AG · Nachrichtenkabel und Systeme · 5201 Brugg
Telefon 056 46 03 100 · Fax 056 46 03 531 · http://www.bruggtc.ch

Leistung, die verbindet