A Constraint-Oriented Service Creation Environment

Bernhard Steffen*, Tiziana Margaria*, Andreas Claßen*, Volker Braun*, Rita Nisius†, Manfred Reitenspieß†

Intelligent Network (IN) services are customized telephone services, like e.g., 1) 'Free-Phone', where the receiver of the call can be billed if some conditions are met, 2) 'Universal Private Telephone', enabling groups of customers to define their own private net within the public net, or 3) 'Partner Lines', where a number of menus leads to the satisfaction of all desires. The realization of these services is quite complex and error prone.

The current trend in advanced IN services clearly evolves towards decoupling Service Processing Systems from the switch network (see e.g. [3]). The reasons for this tendency lie in the growing need for decentralization of the service processing, in the demand for quick customization of the offered services, and in the requirement of rapid availability of the modified or reconfigured services. Service Processing Systems are those elements of the IN architecture which provide *service processing logic and control* but not *connection control and management*, which are provided by the underlying switch system. In particular, they include Service Creation Environments[1], which are interactive environments responsible for the creation, modification, customization and provision of new services.

Service Creation Environments for the creation of IN-services are usually based on classical 'Clipboard-Architecture' environments, where services are graphically constructed, compiled, and successively tested. Two extreme approaches characterize the state of the art: The first approach guarantees consistency, but the creation process is strongly limited in its flexibility to compose Service Independent Building Blocks (SIBs) to new services. The second approach allows flexible compositions of services, but there is little or no feedback on the correctness of the service under creation during the development: the validation is almost entirely located after the design is completed. Thus the resulting test phase is lengthy and costly.

*Universität Passau, Innstr. 33, D-94032 Passau (Germany), tel: +49 851 509.3090, fax: +49 851 509.3092, {steffen,tiziana,classen,v.braun}@fmi.uni-passau.de.

†Siemens Nixdorf Informationssysteme AG, Otto-Hahn-Ring 6, D-81739 Munich (Germany), tel: +49 89 636.42393, fax: +49 89 636.48976, rei@rust.mch.sni.de .

[1]The Service Creation Environment Function is a major component of the IN architecture ([1, 4]).

Our environment is used for the reliable, aspect-driven creation of telephone services in a 'divide and conquer' fashion: initial prototypes are successively modified until each component satisfies the current requirements. The entire service creation process is supported by thematic views that focus on particular aspects of the service under consideration. Moreover, the service creation is constantly accompanied by on-line verification of the validity of the required features and the executability conditions for intermediate prototypes: design decisions that conflict with the constraints and consistency conditions of the intended service are immediately detected via model checking. Thus, in addition to the facilities offered by classical 'Clipboard'-Architectures', our approach is characterized by the following four properties:

- during the entire creation process there exists a current executable prototype that can be tested and animated,
- the support through thematic *views*, which provide the required global context and hide unnecessary details, allows the designer to choose a particular aspect of interest, and to develop and investigate the services under that point of view. This supports a much more focussed service development, which concentrates on the design of the aspect currently under investigation,
- a macro facility allows to define whole subservices as primitive entities, which can be used just as SIBs. Macros may be defined on-line and expanded whenever the interna of a macro become important. This supports a truly hierarchical service construction.
- the *global consistency* of each design step with implementation-related or service-dependent frame conditions is *automatically verified*. Thus sources for typical failures can be immediately detected even in the presence of macros.

Figure 1 summarizes the global structure of our approach, which supports an arbitrary decomposition of the design process. This is necessary, since the same Service Creation Environment is shared by teams of users with completely different profiles. According to [2] at least the following user profiles are envisaged:

- The *service programmer* has advanced programming skills and uses the SCE to create new generic functions,
- The *service designer* has logical skills and uses the above functions to create new services,
- The *service provider* is familiar with the specific customer needs and enters customer specific data into the data files.

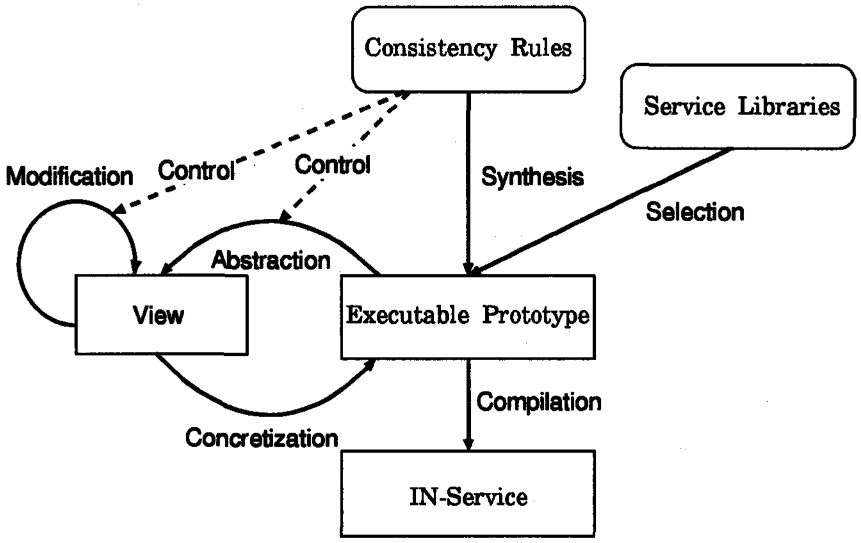

FIGURE 1. The Service Creation Process

We offer the needed design flexibility by means of the second of the following three phases:

1. In a first step, an existing service of similar application profile is loaded from the service library, or a completely new design from scratch is done (under model checking control). Of course this design is supported by the macro facitility. Alternatively, initial executable prototypes could automatically be generated from the set of underlying consistency conditions and constraints, a feature, which is not part of the current version of our service creation environment.

2. The second step consists of aspect-driven modification: the user chooses the aspect of interest, generates the corresponding view abstracting from all irrelevant details, and modifies it where necessary. This is iterated until all relevant aspects have been treated. Due to the on-line verification with the model checker, the executability is preserved and erroneous design steps are immediately detected. It is in this phase, where macros may be required to be expanded in order to resolve 'internal errors'.

3. Current prototypes can at any time be tested, compiled, executed, and, if satisfactory, stored in a data base.

It should be noted that the macro facility covers the standard ***stepwise refinement*** approaches of the usual tools for service creation. In fact, in combination with our concept of views, macros allow an enormously flexible service development. In addition, views support the realization of a very flexible *access control mechanism*, by allowing designers and service providers the definition of customer specific views with restricted modification potential. Views provide in fact a natural division of the central design process (2nd step) into levels. In particular this allows us to tailor the environment for the specific needs of the service designer, the service provider, the customizer, and the user. The view-specific hiding can be used to automatically take care of access permissions.

Implementation

The implementation of the Service Creation Environment is based on METAFrame (see [6]). It is currently available for SUN SparcStations under UNIX, for PCs under LINUX, and for Siemens RM Machines under SINIX. Its graphical interface as well as the hypertext browser are built on top of the Tcl/Tk graphics library [5]. Target language is the HLL, whose interpreter is implemented in C++.

Acknowledgements

We are grateful to Michael von der Beeck, Achim Dannecker, Philipp Florschütz, Carsten Friedrich, Andreas Holzmann, Marion Klein, Dirk Koschützki, Gerald Lüttgen, Falk Schreiber, Markus Schweighofer, and Matthias Seul for their cooperation in the design and implementation of METAFrame . We also thank the Siemens Nixdorf IN team and the GMRS team for their precious interaction in the definition of the characteristics of the system and for their valuable contributions to the realization of the current product.

1 REFERENCES

[1] J. Aitken: "*Intelligent Networks*", Seminar, Logica UK Ltd., London, April 26–27, 1995.

[2] P. K. Bohacek, J. N. White: "*Service Creation: The Real Key to Intelligent Network Revenue*", Proc. Workshop Intelligent Networks '94, Heidelberg, May 24-26, 1994.

[3] E. Crabill, J. Kukla: "*Service Processing Systems for AT&T's Intelligent Network*", AT&T Techn. Journal, Vol.73(6), 1994, pp.39-47.

[4] ITU CS1 Recommendations, 1993.

[5] J.K. Ousterhout: "*Tcl and the Tk Toolkit*," Addison–Wesley, April 1994.

[6] B. Steffen, A. Claßen, T. Margaria: "*The* METAFrame *: An Environment for Flexible Tool Management*," Proc. TAPSOFT'95, Aarhus (DK), May 1995, LNCS N.915, pp.791-792.