Development of New Telecommunications Services in Distributed Platforms: A Structured Approach

Dionisis X. Adamopoulos, George Pavlou
Centre for Communication Systems Research (CCSR)
University of Surrey, Guildford, GU2 7XH
{D.Adamopoulos, G.Pavlou}@eim.surrey.ac.uk

Constantine A. Papandreou
Hellenic Telecommunications Organisation (OTE)
17 Kalliga Street, GR-114 73, Athens, Greece
kospap@org.ote.gr

Abstract - It is broadly accepted that the telecommunications industry is undergoing a fundamental restructuring. This is driven by new customer demands and by changes in the regulatory environment, and is both enabled and encouraged by rapid technological developments. Additionally, the demand for new sophisticated telecommunications services with multimedia characteristics is increasing. These services require more flexible access, management, and charging mechanisms. Thus, it is necessary to assist, in a systematic manner, telecommunications service designers in the development of new services. In this paper, after a brief examination of important service engineering matters, a service creation methodology is proposed and presented focusing on its essential characteristics. A possible enhancement of this methodology through the use of design patterns and frameworks is then considered.

I. INTRODUCTION

The coming years will herald unprecedented changes in the structure and capabilities of telecommunications networks. This evolution and diversification is being driven mainly by liberalization, increasing competition in the marketplace, technological advancements, and demands from all customer segments for an increasingly sophisticated portfolio of telecommunications services, tailored to their specific needs. It is expected that the forthcoming integrated (fixed and mobile) broadband networks will be openly available to existing and new service providers, constituting a worldwide common shared communications platform for a multitude of new advanced telecommunications services (telematic services). The proliferation of these services will lead eventually to the transformation of the global information infrastructure to an open services market fueled by deregulation, strategic partnerships, and joint interoperability activities.

There is much incentive to stay ahead of this global market, and offer new and / or improved services before the competition. Pressure on service providers is increased as they need to be able to react quickly and flexibly to ever changing customer needs by developing and offering services of increasing functionality and diversity in shorter time-frames. Therefore, rapid deployment of new or improved services is critical, and the service life-cycle has to accelerate so that new services can be provided fast enough to meet changing customer demands in a competitive manner. However, the fast and cost-effective provision of new efficient services requires not only an open service architectural framework, like the one specified by the Telecommunications Information Networking Architecture Consortium (TINA-C) [10], but also appropriate support for the service development process [1,6].

In this paper, in order to structure and control the service creation process from requirements capture and analysis to service implementation, to reduce the inherent complexity, and to ensure the thorough compatibility among the many involved tasks, a TINA-C conformant service creation methodology is proposed. Its intention is to provide valuable answers to several important service engineering matters and thus facilitate the transition to a telecommunications environment where many different (enhanced) services are provided by a multiplicity of service providers to several categories of customers within an open market.

II. THE NEED FOR AN INTEGRATED & SYSTEMATIC APPROACH

An architectural framework is by its definition an abstract entity, which consists of a set of concepts / principles and a set of guidelines and rules. For this reason, TINA-C is more descriptive rather than prescriptive, and its application can be a complex task [7]. Furthermore, there seems to be no end to the emergence of new services, each requiring new set of communications capabilities. In a world already replete with a multitude of services, the addition of new intricate services can be a daunting challenge.

The basic factors that shape this challenge are addressed by the discipline of integrated service engineering, which includes the technologies and engineering processes required to define, design, implement, test, deploy, maintain, and manage telecommunications services. The concept of the service life-cycle has a central role in integrated service engineering. All services go through a service life-cycle, which contains descriptions of activities, in the form of an ordered collection of processes or steps, that are required to support the development, the operation, and the maintenance of a service [6,12].

Fig. 1 depicts a graphical representation of an enriched variation of the TINA service life-cycle model. The rectangles are the actions / phases, while the ellipses are the main states that a service goes through (conceived but not planned, planned but not installed, installed but not activated, activated but not instantiated, and instantiated (executing)).

0-7803-6283-7/00/$10.00 © 2000 IEEE

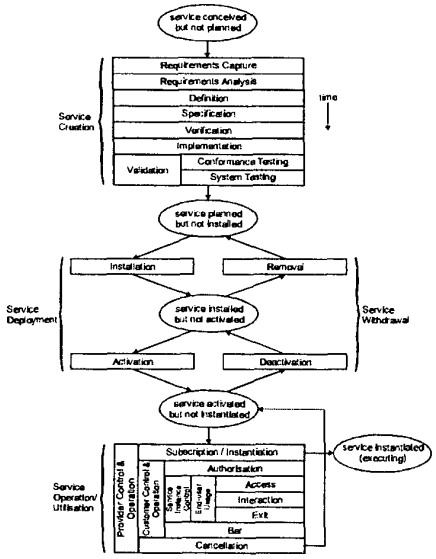

Fig. 1. The service life-cycle.

Among all the stages of the service life-cycle of Fig. 1, in TINA-C, service creation is one of the most abstract and general, since it does not provide many guidelines on how to structure each of its phases. Furthermore, it is also one of the most important as it determines the efficiency with which the services will be developed, and thus the success of service providers in a highly competitive market.

In order to meet these challenges, a service creation methodology is proposed to enable TINA-C to satisfy the required expectations on long-term efficiency of service design, provision, and management. This methodology, given a set of requirements that a service should meet, a set of the available service independent features (normally in the form of service components), and a target TINA-C compliant Distributed Processing Environment (DPE) wherein the service will be deployed, facilitates the design and implementation of a TINA-C compliant service, which meets the desired requirements by promoting the use of the service independent features [3,9].

III. The Proposed Methodology

Telecommunications operators need to master the complexity of service software, because of the highly diversified market demands, and consequently, because of the necessity to quickly and economically develop and introduce a broad range of new services. To achieve such an ambitious, yet strategic to the telecommunications operators goal, a service creation methodology based on the rich conceptual model of TINA-C is proposed.

A high-level or macro-level view of the proposed service creation methodology can be seen in Fig. 2. The proposed service development process is based on an iterative and incremental, use case driven approach. An iterative service creation life cycle is adopted, which is based on successive enlargement and refinement of a telematic service through multiple service development cycles within each one the telematic service grows as it is enriched with new functions. More specifically, after the requirements capture and analysis phase, service development proceeds in a service formation phase, through a series of service development cycles. Each cycle tackles a relatively small set of service requirements, proceeding through service analysis, service design, service implementation and validation, and service testing. The telematic service grows incrementally as each cycle is completed.

According to Fig. 2 the main phases of the proposed methodology are the following:

- *Requirements capture and analysis phase:* It identifies the telematic service requirements (together with a number of roles), and presents them in a structured way.
- *Service analysis phase:* It describes the semantics of the problem domain that the telematic service is designed for. Thus, it identifies the objects that compose a service (information service objects), their types, and their relationships.
- *Service design phase:* It produces the design specifications of the telematic service under examination. Computational modeling is taking place in this phase and thus the service is described in terms of TINA-C computational objects interacting with each other.
- *Service implementation phase:* In this phase the pieces of the service software (computational objects) are defined and implemented in an object-oriented programming language (e.g. C++, Java), inside a TINA-C compliant DPE.
- *Service validation and testing phase:* It subjects the implemented telematic service to a variety of tests in order to ensure its correct and reliable operation.

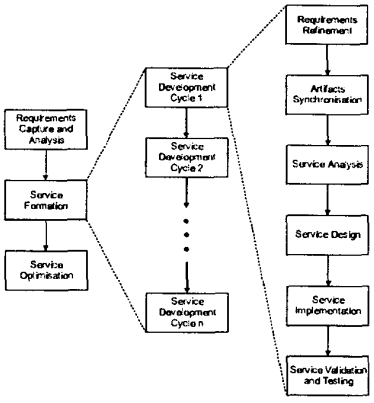

Fig. 2. Overview of the proposed methodology.

- *Service optimization phase:* It examines thoroughly the service code in order to improve its performance in the target DPE, and thus prepare the telematic service for a successful deployment.

As can be seen from Fig. 2, the proposed methodology is conceptually consistent with the viewpoint separation as advocated by TINA-C in accordance with the Reference Model for Open Distributed Processing (RM-ODP), and uses the service life-cycle of Fig. 1 as a roadmap. It has to be stressed that the proposed methodology does not imply a waterfall model in which each activity is done once for the entire set of service requirements. Furthermore, graphical and textual notations are proposed for almost all phases to improve the readability of the related results and ensure a level of formalism sufficient to prevent any ambiguity. In the following paragraphs the most important phases of the proposed methodology are examined. The service optimization phase has been omitted, because it depends highly on the selected programming language and on the target DPE.

A. Requirements Capture and Analysis

During this phase the service developer assembles, documents, and structures the requirements on the service (service needs) from the different stakeholders involved. The focus is on modeling the concepts that are visible at the service boundary, and thus the service logic is viewed as a black box.

One of the most important tasks pertaining requirements capture is the identification of the independent entities / actors, which are involved (by their collaboration) in the operation of the service within and across business administrative domain boundaries. These entities correspond to roles modeling a well defined grouping of functionality under control of a specific stakeholder [12].

A role can be either generic or specific. The main generic (business) roles and the (business) relationships between them are specified by the TINA Business Model [11]. Each generic role (consumer, retailer, broker, third party service provider, connectivity provider) corresponds to one or more specific roles.

TINA-C reference points are defined in relation to the (business) relationships they support, and, according to their functionality, can be divided into access segment and usage segment reference points. This segmentation of reference point definitions enables any inter-domain reference point to be defined with the minimum set of functionality needed for the (business) relationships being analyzed.

In order to proceed to the service analysis phase, the reference point segments should be mapped to feature sets, which are groups of interfaces that expose restricted parts of an information model for manipulation or examination, define the details of interactions between components, and specify levels of functionality inside a service (e.g. basic or multiparty session control). Via the mapping to feature sets, the segments for any particular inter-domain reference point should determine the minimum set of service components (or related functionality) that would be required to conform to this reference point.

B. Service Analysis

The aim of this phase is to determine the functionality needed for satisfying the requirements that were identified in the previous phase, and to define the software architecture of the service implementation. For this reason, the focal point shifts from the service boundary to the internal service structure. The output of this phase is mainly the static view of the internal structure of the service.

The service analysis phase is the first phase of the service creation process, where the service is decomposed into constituent parts (Information Objects, IOs), with the appropriate relationships among them, in an attempt to gain an overall understanding of the service. High level Object Modeling Technique (OMT) diagrams are used to represent the main concepts of the service and their relationships. Additionally, analysis level Message Sequence Chart (MSC) diagrams model interaction among service parts. Alternatively, Unified Modeling Language (UML) static chart diagrams can be used to describe the static structure of the IOs, and UML statechart diagrams to specify the dynamic behavior of significant IOs.

The OMT and MSC models defined so far provide a high level overview and a clear understanding of the service as a whole. They do not focus to the individual IOs. Consequently, a formal modeling syntax is needed that allows the semantics of the information model to be precisely captured and specified. Quasi GDMO and GRM are the formal notations used for this reason.

According to the TINA-C service architecture new services can be realized by enhancing already existing components (e.g. with the use of inheritance) or by defining new ones. TINA-C specifies a set of service independent components that support access session, generic service session, and communication session related functionality. These components can be considered as reusable units in the creation of new services. They may be used in a service implementation as they are, or as the basis for the construction of a service specific component [9].

The exploitation of the TINA-C service independent components in the service analysis phase, and in subsequent phases (depending on the nature of the available component libraries), begins with the selection and reuse of the appropriate service independent functionality. Then, the service dependent segment is developed by exploiting as much as possible the service independent segment. Finally,

the two segments are integrated. This process can be expressed with the following series of steps (fine tuning implies a feedback loop):

- Configure the access session related segment:
  - Select the access session related functionality.
  - Customize (if necessary) the selected access session related functionality.
- Configure the service session related segment:
  - Select the service generic functionality.
  - Customize (if necessary) the selected service generic functionality.
  - Determine the service specific functionality.
  - Develop the service specific functionality.
  - Fine tune the relations between the service generic and the service specific part.
- Fine tune the relations between the access session and the service session segment.
- Configure the communication session related segment:
  - Select the communication session related functionality.
  - Customize (if necessary) the selected communication session related functionality.
- Fine tune the relations between the service session and the communication session segment.
- Integrate all three segments (access, service, and communication session).
- Prepare the end user system:
  - Develop an end-point of the service session.
  - Develop / customize an end-point of the access session.

C. Service Design

During this phase the service developer defines the interfaces and the behavior of the IOs that were identified in the service analysis phase, and structures the service in terms of interacting computational objects (service components). The output of this phase is the dynamic view of the internal structure of the service.

As a first step in this phase, the IOs that comprise the service are considered as potential candidates for Computational Objects (COs). In many cases, IOs are mapped to one corresponding CO encapsulating the information defined by the IO and providing an operational interface to access the information. However, the mapping between IOs and COs is not necessarily one to one.

COs are specified using the TINA-C Object Definition Language(ODL). Additionally UML collaboration, sequence, activity, and component diagrams can be used. While the collaboration diagram shows the interactions among object instances and their links to each other, the sequence diagram describes object interactions arranged in a time sequence. The activity diagram allows to specify in which order activities (such as operations provided by a CO interface) have to be executed. Finally, the component diagram shows the organizations and dependencies among components.

D. Service Implementation

During this phase an implementation is generated from the service specifications, and the deployability of the overall implementation on a TINA-C compliant DPE is examined (DPE targeting).

The engineering representation of a CO (using an object-oriented programming language like C++ or Java) is called an engineering Computational Object (eCO). The mapping between COs and their eCOs is one to one: no eCO represents a composition of COs nor is a CO represented by more than one eCOs. The interfaces of an eCO represent the interfaces of its corresponding CO [4].

In this phase, the UML component and deployment diagrams can be used to define mechanisms and functions required to support distributed interactions between service objects in the target DPE. In general, the use of the same UML notation in almost all the phases of the methodology, has the advantage of making both the service description more coherent and the process of proceeding from one phase to another more natural and efficient. For this reason, TINA-C could also consider UML in a future version of its computing and service architectures.

E. Service Validation and Testing

Validation takes place in this phase by comparing the developed service software against the service specifications produced at the service design phase. This activity can be subdivided into the following two subactivities:

- *Conformance testing:* It involves checking the implementation for conformance to architectural rules and standards used in the service design.
- *System testing:* It comprises the testing of service software in a (possible) operational environment.

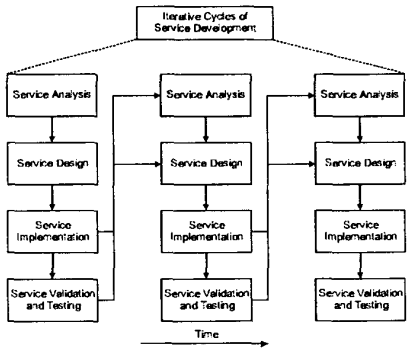

Fig. 4. The influence of service implementation work in the proposed service development process.

With this phase a service development cycle ends and another one (depending on the exact nature of the specific service requirements) is ready to start. A significant strength of the iterative and incremental character of the proposed methodology is that the results of a prior service development cycle can feed into the beginning of the next cycle. Thus, subsequent service analysis and service design results are continually being refined and informed from prior service implementation work (see Fig. 4). For example, when the code in cycle N deviates from the service design of cycle N (which it inevitably will), the final service design based on the implementation can be input into the service analysis and service design models of cycle N+1. For this reason, as can be seen from Fig. 2, an early activity within a service development cycle is to synchronize the created artifacts. The artifacts of cycle N will not match the final service code of cycle N, and they need to be synchronized before extended with new service analysis and design results.

IV. Exploitation of Design Patterns and Frameworks

A typical telematic service is a large scale system. For this reason, the successful application of the proposed service creation methodology is a rather complicated task, where the effective and efficient communication of architectural knowledge between the service designers and developers is of great importance. To facilitate this communication the exploitation of design patterns and frameworks in the service engineering area is suggested.

In the case of TINA-C, design patterns can be defined by identifying groups of interworking service objects, where every group is characterized by a micro-architecture that determines the way the objects interact to provide a solution for the specific aspects of a subproblem that arises during the development of a telematic service. Furthermore, a framework can be defined as the overall architecture, which specifies how the identified configurations of service objects can collaborate to implement a solution for the whole problem. Thus, a framework is a kind of construction kit for complete or semi-complete telematic services. It has to be complemented and customized using inheritance techniques [2].

The introduction of design patterns and frameworks in the proposed methodology implies the establishment of a common vocabulary and the definition of common design structures for all persons involved. They assist to reduce the scope of the problem solving process in the case of service creation, because they support the identification of similar problems and similar solutions. However, design patterns and frameworks are abstract concepts. There is no guarantee that their usage will lead to design reusability, design portability, and abstract customizability. Furthermore, good design patterns and frameworks, like good inheritance hierarchies, can not be invented in an easy way. They have to be chosen and designed very carefully [5].

V. Conclusions and Future Work

Explosive increase in service variety as well as in globalization and customization of services produce ever increasing pressures for the efficient support of the service engineering activities in an environment open to changes in market and technology. For this reason, in this paper, the key features that should characterize an advanced service creation and design methodology were presented and examined.

The proposed methodology is being applied to the development of a MultiMedia Conferencing Service for Education and Training (MMCS-ET) using Microsoft's COM/DCOM as a DPE [8]. More specifically, a variety of scenarios are considered involving the support of session management requirements (session establishment, modification, suspension, resumption, and shutdown), interaction requirements (audio / video, text, and file communication), and collaboration support requirements (chat facility, file exchange facility, and voting). The results obtained so far provide confidence that this methodology is useful for the description and development of (complex) new telecommunications services. However, further experience on the application of the methodology is necessary, together with the establishment of a collection of evaluation criteria and metrics for more comprehensive verification procedures.

Under these conditions the proposed service creation methodology is expected to enrich TINA-C and enforce it to pave the way towards an open integrated broadband telematic infrastructure populated by a virtually limitless variety of telematic services.

References

[1] D.X. Adamopoulos, G. Haramis, and C.A. Papandreou, "Rapid prototyping of new telecommunications services: A procedural approach", *Computer Communications*, vol. 21, pp. 211-219, 1998.

[2] K.P. Eckert and P. Schoo, "Engineering frameworks: A prerequisite for the design and implementation of distributed enterprise objects", Proceedings of *EDOC '97*, pp. 170-181, October 1997.

[3] S. Efremidis, D. Prevedourou, L. Demounem, K. Milsted, and H. Zuidweg, "TINA-oriented service engineering support to service composition and federation", Proceedings of *IS&N '98*, LNCS, vol. 1430, Springer-Verlag, Berlin, pp. 409-422, 1998.

[4] E. Kelly, N. Mercouroff, and P. Graubmann, "TINA-C DPE architecture and tools", Proceedings of *TINA '95*, pp. 39-54, February 1995.

[5] E. Koerner, "Patterns for constructing CSCW applications in TINA", Proc. of *IDMS '97*, LNCS, vol. 1309, Springer-Verlag, pp. 322-329, 1997.

[6] P.P. Demestichas et al., "Issues in service creation for open distributed processing environments", Proceedings of *ICC '99*, June 1999.

[7] S. Rana and E. Sellin, "Implementation of a pan-European TINA-compliant service management platform", *Computing & Control Engineering Journal*, vol. 10, pp. 73-78, April 1999.

[8] C.A. Papandreou and D. X. Adamopoulos, "Design of an interactive teletraining system", *BT Engineering*, vol. 17, pp. 175-181, 1998.

[9] N.D. Polydorou et al., "Efficient creation and deployment of telecommunication services in heterogeneous distributed processing environments", Proceedings of *IEEE/IEE ICT '98*, vol. IV, pp. 336-340, June 1998.

[10] TINA-C, "Definition of Service Architecture", Version 5.0, 1997.

[11] TINA-C, "TINA Business Model and Reference Points 4.0", 1997.

[12] T. Mota, P. Hellemans, and T. Tiropanis, "TINA as a virtual market place for telecommunication and information services: The VITAL experiment", Proceedings of *TINA '99*, April 1999.