Systems Development for Service Provider Networks

Ralf Klischewski, Ingrid Wetzel
Hamburg University, Department for Informatics, Software Engineering Group
Vogt-Kölln-Straße 30, 22527 Hamburg, Germany
{klischewski, wetzel}@informatik.uni-hamburg.de

Abstract

In a service-oriented society, service provider networks are promising as they enable the delivery of coordinated comprehensive cross-organizational services. Following strategic considerations from business networking and CRM we focus on system support for complex service processes. As solutions based on highly integrated systems do not seem suitable, we suggest with serviceflow management a new approach for systems development within service provider networks. It provides (1) an application-oriented serviceflow modeling technique for fostering agreement and standardization efforts among providers with respect to flexible service delivery, (2) a minimal "light-weight" generic solution for exchanging serviceflow process representations across providers throughout the network and (3) a set of generic components to fit in a four-level servicepoint architecture for realizing web-interfaces and/or provider work places. The approach leaves open whether providers will use the generic servicepoint architecture or realize the servicepoint functionality separately as long as – as a basis – the commitment to a particular serviceflow model is given and the serviceflow process representation exchange is assured. The article ends up with raising various open questions concerning the organizational as well as the technical potentials of the approach based on serviceflow experiences in the e-government and e-health domain.

Keywords

Business Networking, Information Systems Development, Serviceflow Modeling, Serviceflow Management, Process Management, Generic Software Components

1 Introduction

Today, business strategies such as business networking and CRM [12] aim at finding adequate responses to a fast changing global market with more and more exigent customers. A careful chosen, even flexible net of service providers seems as a promising strategy in that it allows for enhancement and specialization in core competence of the own enterprise while at the same time enabling the provision of comprehensive services as a joint effort in order to gain for maximum market shares. In the search for adequate system support for enabling and designing these strategies it comes to the surface that, possibly, even new development approaches are required. Existing approaches such as ERP or workflow management seem inappropriate as they are based on a significant assumption: the basis of a highly integrated solution. Reasons for "abandoning" this basic presupposition are manifold:

- Business networks comprise the ease of switching partners. Accordingly, the effort for joining in must be as low as possible. Integration requiring large efforts on an organizational as well as a technical level (and time) contradicts this requirement.
- Although network partners need a certain amount of trust (not each aspect of the cooperation can be ensured by contracts and formal agreement [1]), this need not imply the willingness to open up information and enterprise internal knowledge among the partners as it might be inescapable with integrated solutions.
- Fast changing technological settings as present in web-based and distributed programming question generally investments for long-term solutions.
- Standards for integration are slow to be provided and accepted. Thus, it is a challenge to support close cooperation based on a minimal set of agreed standards.

Facing to desert "familiar" paths in the development of systems support, the question is what comes instead. While trying to provide new concepts it seems wise to keep as much as possible from the "goodies" of the past. In section 2, we therefore look briefly at the general approaching and the major intentions of successful system development in general. Based on these findings, we position serviceflow management as parts of a possible business strategy for service provider networks being based on a respective systems development approach and we discuss in to what extent the latter has to deviate from well known approaches (third section). A forth section provides manifold open questions of this approach based on experiences in the e-government and e-health sector.

2 The Role of Modeling Between Application and Implementation

A brief look back on successful contributions in system development exhibits a broad alignment in intention as well as ways to proceed.

In the seventies and eighties, traditional IS development was centered around database applications. A database management system (DBMS) provides as programming interface a data model (a higher level language for definition and manipulation of data) while hiding the specifics of dealing with bulk data and at the same time providing additional functionality such as concepts for concurrency and recovery. The general way to proceed was as follows: Often, an even semantic richer (conceptual/semantic data model, later object oriented) data model than the one provided by the DBMS was used for modeling the application data. The resulting data model was (sometimes automatically) transformed into the executable model on top of which the application was programmed (e.g. [13] [2]. Thus, the overall intention was to provide a modeling approach close enough to easily model the phenomenon of the application world and being at the same time executable.

In the nineties, the focus moved to process instead of data integration. Here, the approaches between modeling and implementation were more independent. Modeling approaches like BPR tried to solve the redesign of processes from a business perspective while domain specific COTS or workflow management systems were provided to implement the redesigned processes. Even though the gap between e.g. BPR and workflow management was seldom addressed in a satisfying way, the intention was similar to the database approaches: provision of executable models which are “close” to capturing real world processes and added generic system functionality such as generic in-box interfaces, monitoring and evaluation devices, hiding of client-server middleware, etc[5][6]. We see the following common points. The approaches provide

1. conceptual modeling which aims at capturing the concepts of the application domain in a “natural” and semantically rich way,
2. transformation rules or assisting tools for transforming the conceptual models into executable ones,
3. management systems making the models executable while adding further generic support, (COTS being more domain dependent allow for fine-tuning by parameterizing the application, sometimes based on customized domain-specific models).

In the context of service provider networking some of the underlying assumptions in these approaches are no longer valid.

Starting with the first point and focussing on common process or workflow modeling, the strictness of a model in the sense of setting rules for process execution (later being enforced by a system) does not match with the general character of service delivery. Due to the "indistinct" dimension of customer satisfaction, the overall approach should enable degrees of flexibility for allowing individualized service provision according to the customer's need. Therefore, our intention regarding the utility of models and hence the modeling activity itself differs in the sense that it explicitly addresses the gap between a serviceflow model and its use context. Accordingly, a serviceflow model defines a standard pattern of performance which (later) "only" serves as a resource for initializing and representing an individualized serviceflow. These individualized representations direct the proper exchange of process knowledge among providers and serve as a cooperation material for expressing and distributing deviations from the pattern thus allowing for the required flexibility in joint service delivery. In correspondence, the respective generic system components are assumed to provide exactly the same handling as assumed during modeling, i.e. support for initializing and changing individualized process representations based on the modeled pattern and support for their exchange.

Considering the second point, a close correspondence between modeling and realization level seems to be preferable especially in the context of distributed providers as it allows the agreement achieved on an inter-organizational level to be still visible and retraceable on the realization level. On the other hand, "locating" the model in implementation needs to be independent from the realizing systems (due to the heterogeneity of technical environments and solutions).

This leads up to point three. As already discussed above, a central, highly integrated management system as realization approach has to be substituted by system components which might be distributed among providers. However, even under this assumption, suitable additional generic services should likewise be provided. They would enable quick joining in of new providers into the network by reducing efforts and investments into technology.

All in all, out of these considerations we conclude for a business strategy based system development approach for service provider networks that has to address the mentioned specifics

- of service provider networks: there is *no centralized serviceflow control* to be assumed. Instead, providers enroll to a network which provides networking components comprising knowledge, software components and organizational agreement.
- of services: serviceflow support should allow *flexible performance* being captured in the modeling approach and being supported by the generic realization in question.

- of traditional approved methods in IS development: a close *relationship between conceptual modeling and implementation* is intended which provides additional *generic services*.

3 Developing Management Systems for Service Provider Networks

Reflecting business strategies in the area of service provider networks (see above) serviceflow management [10][8] aims at (1) providing an easy-to-use modeling language which allows for capturing the specifics of service processes and for fostering agreement and standardization efforts among providers, (2) a "light-weight" generic technical solution for easily enabling a uniform serviceflow connection between providers and (3) a set of generic components to fit in a four-level architecture for realizing web-interfaces at servicepoints.

3.1 Serviceflow Modeling

In general, conceptual modeling calls for providing a way of modeling close to the conceptions of the application domain. In serviceflow management, we all the more need an easy-to-grasp modeling approach since (similar to BPR) it is the modeling which forms the basis for agreement between the providers as well as between providers and customers.

Serviceflow modeling is grounded in a twofold perspective on service processes: considering the relationship nature of services [3] as well as the necessity of their efficient performance. (For a background and details see the e-health and e-government cases in the appendix). In case of service processes, the success of the service (in terms of business revenue) is crucially dependent on the customer/client's experiencing all (even individually adapted) sub-services as coherent and continuous and as parts of a comprehensive overall serviceflow. From the service provider's perspective, it requires the integration and coherence of all situated sub-services across temporal, spatial and team boundaries.

Conceptually and to simplify matters enabling structuring from the provider's point of view, we define serviceflow in terms of servicepoints. Services always create some social situations, it need "places" [4] which frame the situation where service tasks are carried out, e.g.

- service staff evaluating the client's concern and serving his/her needs (in these situations the client's presence may vary from being physical there, being present through telecommunication, being present through one of his/her objects, or only by some sort of the concern's representation),

- client is being served by some autooperational device (e.g. a web portal) on behalf of the service provider.

These places we call service points, and the subsequent interrelation of a number of service points is a serviceflow, which are modeled as indicated in figure 1.

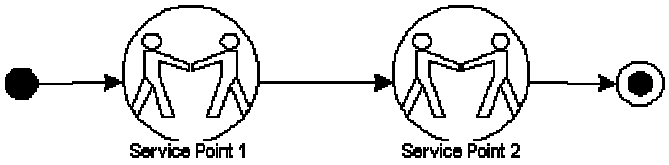

Fig. 1. Serviceflow pattern with two service points

Each service point captures the specific service tasks to be carried out and their respective pre- and postconditions from the provider's point of view (figure 2). For modeling service tasks, we are using UML use cases. Jacobsen ([7], p. 129) defines a use case as a "behaviorally related sequence of transactions in a dialogue with the system." Within serviceflow modeling, these service tasks either refer to a set of activities carried out by service staff with support by an IT system (indicated in the model by aligning service staff to the task) or to a set of operations carried out by the IT system, possibly in dialogue with the client (indicated in the model by aligning the client to the task).

The servicepoint sequence and the respective pre- and postconditions represent the contract for interrelating the service points. As the modeling captures standard procedures, it does not mean that deviations are impossible during execution (s. below). We even introduce ways to distinguish between mandatory and optional conditions and tasks.

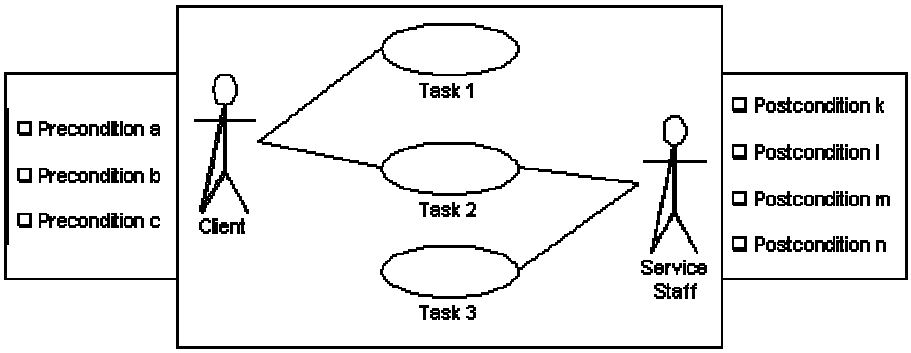

Fig. 2. Servicepoint with three service tasks (with alignments to client service staff) and respective pre- and postconditions

[6]

For capturing more application knowledge, each service point activity should be linked to a rich description (e.g. scenario) and a glossary. Cooperation pictures [11] can also augment the serviceflow representation to explicate recurrent cooperation relations among the involved actors. To construct an enriched serviceflow model the resulting documents of all these techniques should be related for display, e.g. by interlinking HTML version of the different documents.

Overall, modeling of serviceflows in terms of servicepoints allows for "modularizing" complex processes in "natural" ways (overall process with sub-processes at servicepoints). By this, the modeling process is broken into manageable parts where each provider organization might be responsible for modeling their own servicepoints (or a central modeling group might be in charge). Finally, the interrelation of servicespoints will be subject of consistency checks, negotiations and reorganization between service providers concerning e.g. matching pre- and postconditions or the distribution of tasks among the service points.

The serviceflow patterns capture "best practice" (i.e. efficient performance) and serve as a contract among providers at "design" time. However, at "service provision" time (run time), personalization of these patterns is required as well as the ability for deviations in individual flows. It might include, that e.g. tasks may be carried out although some preconditions are not true, tasks may differ from the pattern although all preconditions are true, or postconditions may not be achieved although tasks have been carried out according to the pattern (or the other way around). Furthermore, the interrelation of service points itself should be subject to possible changes as well: whereas the serviceflow history (the sequence of passed service points) is, of course, not changeable, the serviceflow schedule, i.e. the part of the serviceflow with service points not visited yet, may be manipulated by deleting or adding service points or changing the sequence order. While some indicators for flexibility can be expressed in the model (e.g. mandatory or optional conditions or tasks), ad-hoc deviations from the pattern are mainly a question for adequate system support.

3.2 The "Light-Weight" Solution: Exchanging Servicefloats

The implementation is centered around the technical representation of the modeled pattern – the servicefloat master. During the initialization of a new serviceflow, this pattern gets personalized and serves in the beginning as a standard process plan. During service delivery the personalized servicefloat "mutates" from being a plan into having a dual nature comprising the growing process history documenting already visited servicepoints and a shrinking planning section determining to-be servicepoints in future (which might follow or deviate from the original pattern) being separated by the current servicepoint. These servicefloats

capturing personalized, constantly up-to-date process knowledge get exchanged between serviceflow providers. The approach bears the following potentials:

- Initializing a servicefloat by copying (and possibly adapting) a standard serviceflow pattern guides each provider how to deliver the service in case of a standard performance.
- Documenting the history enables a service provider at a later servicepoint to be informed about the history of the service, about deviations from the standard (e.g. reordered sequence of servicepoints, relaxed preconditions) and about reasons for deviating.
- The constant update of servicefloats determining the current and next servicepoints forms a basis for automating the delivery of servicefloats to the next provider.
- Enabling providers to access and update the process representations (as a material) allows for flexiblity and instant realization of changes during service provision.

In order to exchange servicefloats between provider organizations, they are technically represented as XML documents. Accordingly, a "light-weight" requires merely the distribution of XML DTDs at each provider's site, a facility to exchange XML servicefloats and the provision of master servicefloats for initialization at the first servicepoint. Furthermore, the choice of actual widespread technology together with standard tools for XML handling seems to be acceptable for many providers and facilitates a very simple form of integration in heterogeneous technological settings.

However, successful cooperation in service processes through the exchange of servicefloats needs to comprise more than a mere data representation being sent back and forth. Similar to the notion of a protocol (including both standard representations as interface and rules for handling) it requires an among all parties accepted and assured conventions how to update the servicefloat (i.e. updating it according to the performed tasks, appending the current servicepoint to the history and setting the next servicepoint to the current). Intentionally, serviceflow management leaves it open how each provider realizes the convention. Nevertheless, our approach offers a generic Java component (s. next section), which can be used.

Summarizing, the "light-weight" solution for simply connecting providers requires merely the exchange of XML servicefloats and the keeping of conventions to update it (s. next subsection for ready-to-use components).

3.3 Servicepoint Architectures

The "light-weight" solution offers not much more than a connecting facility between providers. Concerning the original support of diversified service tasks at each servicepoint, it

neither indicates nor determines how this support is accomplished. The realization is left open to each provider's technological infrastructure and degree of intended integration. It can range from using a legacy system to providing a web-interface for providers or client-provider communication.

Nevertheless, with serviceflow management we want to provide additional generic support. The servicepoint architecture provides a generic servicepoint interface for flexibly performing service tasks and allows for integrating tools for task performance. Similar to the servicefloat, the "service knowledge" at each servicepoint (list of tasks and their pre- and postconditions) is represented in a so-called servicepointscript represented as XML-document. A servicepoint manager is responsible for updating both the servicepointscript and the servicefloat and for the reception, delivery and update of servicefloats.

The client-server architecture underlying the servicepoint component has three server layers as indicated in figure 3:

1. Frontend: client to present the user interface
2. Interaction: server layer to organize the user dialogue
3. Serviceflow application: server layer to realize the XML document processing including the servicefloat manager
4. Persistence: the server's file system or date base for saving and retrieving XML documents

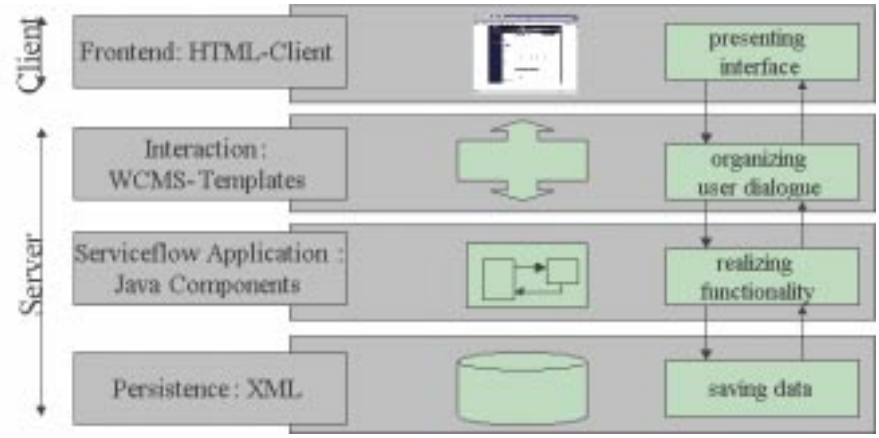

Figure 3: IT Architecture for a Web-Based Servicepoint

In one of our prototypes, the user dialogue is built using Java Server Pages technology whereas in the other, an integration with templates created in a web content management

system (WCMS) is realized. Both dialogue implementations include Java method calls addressing the public interface of the underlying serviceflow application layer implemented in Java.

The components of serviceflow application layer[1] encapsulate the processing of the XML documents related to serviceflow management. On the basis of the XML DOMs servicefloat and servicepointscript objects are created providing a variety of get- and set-methods the interaction layer calls on. More detailed, the servicepoint manager includes methods for retrieving the relevant XML files, creating Document Object Models (DOM) of servicefloat and servicepointscript for a specific customer, saving the manipulated DOMs in XML files and preparing the servicefloat for dispatch (servicefloat manager). In our preliminary version, the persistence layer was implemented using simple file system support while future work will include the usage of XML databases.

3.4 Generic Support for Service Provider Networks

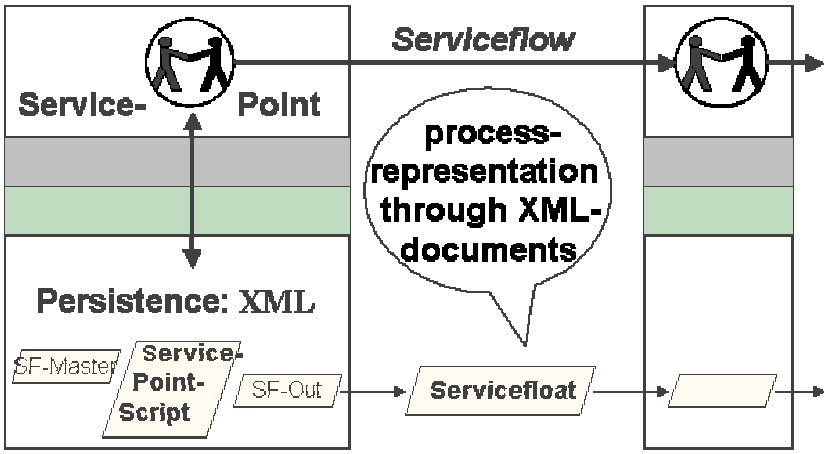

Figure 4: Servicefloats and Servicepointscripts

[1] The description is based on the prototype developed at Hamburg University as presented by the end of January 2001. The technology transfer to project partners is ongoing.

Generic support for service provider networks is oriented around our serviceflow modeling approach focussing on process pattern (indicated top level in figure 4). A serviceflow is understood as a sequence of servicepoints where each servicepoint is further specified by a list of tasks and a list of pre- and postconditions. Generic system components are offered in two stages where due to the autonomy of providers it is left open whether providers will use the generic components at all or realize the system functionality individually. The lightweight solutions offers the servicefloat, an XML-representation of a serviceflow model determining the sequence of servicepoints without incorporating details concerning servicepoints. Together with conventions how to send, receive and update it, servicefloats forms a minimal contract among providers enabling flexible joint service delivery (indicated bottom level in figure 4). The direct correspondence between the modeled pattern (level 1) and its technical representation (level two) is obvious. With the servicepoint architecture, components are provided which support the actual task performance, enable the ability to alter planned processes, tasks and conditions and provide a user interface (indicated virtual layer in figure 4).

Both stages of generic support already indicate the suitability of this approach in order to quickly connect service providers despite of their heterogeneity in both, the organizational and the technical sector. Applying serviceflow management the system development process comprises the following steps:

1. Agreement upon standard processes which are captured in serviceflow pattern. This includes a commitment in offering preconcerted services at each servicepoint. Often the design will follow the existing way of service delivery.
2. Thinking about innovations in serviceflow patterns, in particular how servicepoints which require physical encounter of provider and customer might be transformed to web-based meetings points.
3. Distribution of (possibly adapted) DTDs for servicefloats and servicepointscripts.
4. Commitment to use the exchange mechanism for process representation belonging to serviceflow management.
5. Consideration whether to use the generic components for realizing provider workplace solutions or not and how to integrate the system with existing infrastructure in the provider organization.

4 Open Questions and Future Directions

Serviceflow management is presented as an example for business strategy based software engineering in the context of service provider networks. Up to now, it is based on prototypes,

project experiences and already successful conceptual and implementation concepts. We are convinced that service processes in the context of business networking and CRM require specific support such as the ones introduced here. However, there are a lot of open questions:

- The main question is whether the underlying concept suffices to enable flexible networks as well as to provide satisfying generic support for service providers.
- Our experience lies in the public sector with examples from e-government and e-health. The question arises to what extent the concept will be applicable to a broader set of application areas including commercial service provision.
- With increasing project experience we evaluate whether the concept is simple enough to form a basis of agreement within e-service provider networks. The question addresses both, the style of modeling (for which we received already promising feed-back) and the provided generic components. They might serve as a basis for providing a technical solution for a great variety of e.g. e-government processes.
- New questions arise from the development process itself, e.g.: Who has a central integrating position and which kind of additional committees and commitments are necessary [9]? How can the redesign of service processes be systematically supported (e.g. innovation trough virtual service points)? What kind of contracts among providers are required and whether a published "model space" forms a sufficient commitment?
- Besides serviceflow management, many complex services require the accessibility of documents by multiple parties throughout the whole process (s. examples in appendix A1.1). We therefore distinguish an information flow (described in this paper) versus an information systems approach. The information systems perspective approaches again towards an integrated solution (or a combination of both), possibly being provided from an (ASP) application service provider (an promising concept in e-health). With our e-health prototype we are considering to what extent this affects the provided generic components.
- There does exist a large amount of ideas regarding necessary extensions of the concept and redesign of the generic components. On a conceptual level, subjects are: Support of parallel involvement in several processes, extension of expressability in modeling, esp. concerning flow expressing, separation of process knowledge from data, esp. in case of huge amounts of data and their cross-process usage. Concerning technology, we address: redesign of generic components and integration issues (allowing for more fine-grain stages in support and the integration of webcontent management systems and legacy systems). In regard to user interface design: improved facility for comparing a standard with an individual serviceflow and extended support for defining deviations.

References

[1] Alt, R., Puschmann, T. Reichmayr, C.: Strategies for Business Networking, in Hubert Österle, Elgar Fleisch, Rainer Alt (eds): Business Networking – Schaping Collaboration Between Enterprises, Springer-Verlag, 2001, pp. 89-110.

[2] Borgida, A. Mylopoulos, J., Schmidt, J.W., Wetzel, I.: Support for Data-Intensive Applications: Conceptual Design and Software Development. *In Proceedings of the Second International Workshop on Database Programming Languages*, Portland, Oregon, June 1989.

[3] Gutek, B.: *The Dynamics of Service*, San Francisco: Jossey-Bass Publishers, 1995.

[4] Harrison, S., Dourish, P.: Re-Place-ing Space: The Roles of Place and Space in Collaborative Systems, *Proceedings CSCW'96*, pp. 67-76.

[5] Jablonski, S., Bußler, C., *Workflow-Management: Modeling Concepts, Architecture and Implementation*, International Thomson Computer Press, 1996.

[6] Jablonski, S., Böhm, M. Schulze, W., *Workflow-Management, Entwicklung von Anwendungen und Systemen*, dpunkt.verlag,1997.

[7] Jacobson, I. (et al.): *Object Oriented Software Engineering. A Use Case driven Approach*. ACM Press, 1992

[8] Kaschek, R., Klischewski, R., Wetzel, I.: A Virtual Debate on Serviceflows. *EMISA Forum* 1/2001, pp. 16-23

[9] Klischewski, R., Systems Development as Networking, in: H. M. Chung (ed.): *Proceedings of the 2000 Americas Conference on Information Systems* (August 10-13, Long Beach, CA). Association for Information Systems 2000, p. 1638-1644, 2000.

[10] Klischewski, R., Wetzel, I., Serviceflow Management, *Informatik Spektrum*. 23 (1), February 2000, pp. 38-46.

[11] Krabbel, A., Wetzel, I., Ratuski, S.: Participation of Heterogeneous User Groups: Providing an Integrated Hospital Information System, In: *Proceedings of the Participatory Design Conference* (PDC'96), Cambridge, Massachusetts, USA, pp. 241 – 250, 1996.

[12] Österle, H., Fleisch, E., Alt, R.: *Business networking. Shaping enterprise relationships on the Internet.* Berlin: Springer, 2000

[13] Teorey, T.J., Yang, D., Fry, J.P.: A logical design methodology for relational databases using the extended entity relationship model. *ACM Computing Surveys*, 18(2): 197-222, June 1986.

Appendix

Both cases presented (from e-health and e-government) origin from long-term contacts and were addressed in a teaching project at the University of Hamburg during the winter semester 2000/1. The cases were used to explore our modeling and realization approach for serviceflow management. We performed a requirements analysis in external organizations and presented the prototypes to the external audience. As a result, the approach is applied in the context of realizing the postal vote application through www.hamburg.de for which we expect evaluations already up to autumn 2001. They might serve as a basis for providing a technical solution for a great variety of e-government processes in the city state of Hamburg. In the e-health area, the work is ongoing as well and a network of future project partners is under search.

A 1 Seriveflow Case from E-Health: Preparation, Performance and Aftercare of a Stationary Operation

The following scenario for the preparation, performance and aftercare of a stationary operation is given as an example for a complex serviceflow. The description is oriented around servicepoints where service provider and patient meet, and the relations between them. As an overall picture, figure 5 visualizes different servicepoints related to the involved providers between which the patient and related documents move back and forth.

A1.1 A scenario for the preparation, performance and aftercare of a stationary operation

Usually, if a person faces health problems he/she consults his/her family doctor. If the patient's case seems to be severe or dubious or if the patient is in the need of special treatment as in the case of ongoing pain in the hip, the family doctor transfers the patient with a referral slip to a specialist, here an orthopedist. If the specialist's investigations result in the need for an operation – whereby the specialist might try to achieve a more comprehensive picture by involving additional specialists of other fields – the specialist sends the patient to a hospital. Often, the choice of the hospital depends on the specialist's recommendation or more seldom on the patient's preference.

The patient receives again a referral slip which includes the indication and a short diagnoses. With this form the patient goes to the chosen hospital, usually during open consultation hours or he might make an appointment beforehand. In the investigated

specialized clinic, the patient undergoes a further short investigation in which the clinic physician controls and specializes the indication. After this, the patient is sent to the registration where a date for an operation has to be determined (sometimes including the choice of a certain physician) and preparations such as blood donor or in case of allergies specific implant tests have to be planned. Additionally, the clinic could require further x-ray pictures etc. With a list of preparing activities some of which are already made concrete with an appointment at a certain service provider the patient is sent home. Passing the appointments one by one, the patient might receive certain documents for delivery to the hospital whereby some results might instead be sent by mail, depending on the kind of documents or the arrangements among the providers.

At the patient's arrival at the hospital usually a day ahead of the operation, his/her status of preparation gets controlled by the admission and, if all requirements are fulfilled, the patient gets admitted for the operation. Meanwhile, the results of further investigations hopefully arrive within the hospital at the physicians/surgeon in charge.

After the patient's operation, stationary treatment and finally dismissing, the specialist or family doctor get informed through a physician letter the arrival of which unfortunately usually lasts several weeks. In this case, the specialist has to start the aftercare treatment before he is sufficiently informed about the patient's treatment during the hospital stay.

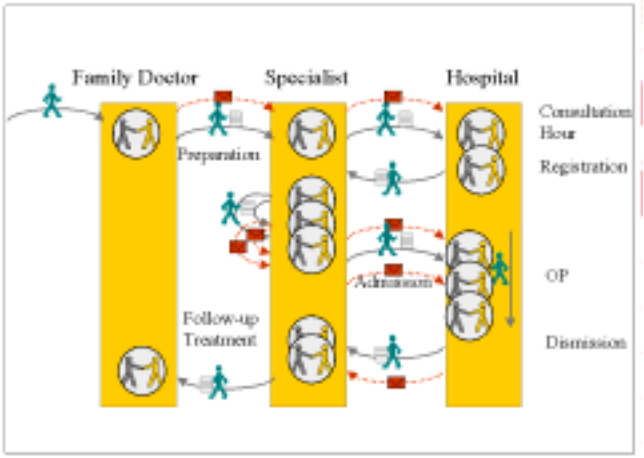

Figure 5: Service Example: Preparation, Performance and Aftercare of an OP

A1.2 Problems and Different Perspectives

As one can easily recognize in the above outlined service process, a number of problems are at hand which easily affect the performance of the process and demands for flexibility and exchange of process knowledge among service providers:

Responsibility in planning

It is striking to consider that no one of the providers does have the responsibility of planning the process. It is even hard to tell who could be in a position to do so. Rather, it looks like that the service process is dependent on the ability of each involved provider to interfere or add to usual procedure according to situated insights about the patient's case or response to treatment.

Overview over process status

Another difficulty lies in the fact that no involved provider does have any overview over the complete picture of the actual status and evolvement in the process. This becomes obvious in the case of aftercare where the responsible specialist might not have received any information while being in the need of starting the treatment.

Situatedness of the process

Special consequences are at hand in regard to the obvious necessity to deliver the service according to situated needs. Each of the involved parties should be informed or at least should be able to retrace the reasons for deviations from tacitly assumed ways to proceed.

Documents in time

A further problem relates to the exchange of documents. Today, we have to face an absence of clear rules. It is neither clear in which way documents are to be delivered nor it is fixed which kind of documents should be exchanged at all.

A.1.3 A Work Scenario with the Serviceflow Prototype

The prototype was developed on the basis of the presented SFM architecture. Servicefloats capture the actual individual history of a serviceflow and planned appointments or activities while servicepointscripts represent lists of standard activities and pre- and postconditions. In our first prototype we applied an IS-oriented realization (s. section 3) storing and manipulation serviceflows, servicepoints and data from different providers on a central application server. This approach was chosen since today many of the involved service provider in the health sector are very poorly IT equipped. Future scenarios will consider the highly heterogeneous infrastructure reaching from poor IT support at physicians to large hospitals using high sophisticated IT landscapes with hospital information systems integrating a huge variety of different specific systems and web servers. So far, different alternatives are at the horizon.

The web-based user interface at a service point provides access to the underlying process information (servicefloats and servicepointscripts) as well as to exchanged data. It has four main section, as indicated in figure 6.

- Section 1 represents the provider/network providing space for its corporate identity.
- Section 2 visualizes the actual serviceflow for the patient in question.
- Section 3 provides the list of activities to be carried out at a chosen servicepoint.
- Section 4 captures the patient record represented as a folder with forms and documents about patient's treatment in the SF. In case of starting the system/choosing a service point without having chosen a certain patient's case, this area presents an overview over all patients/at this servicepoint.

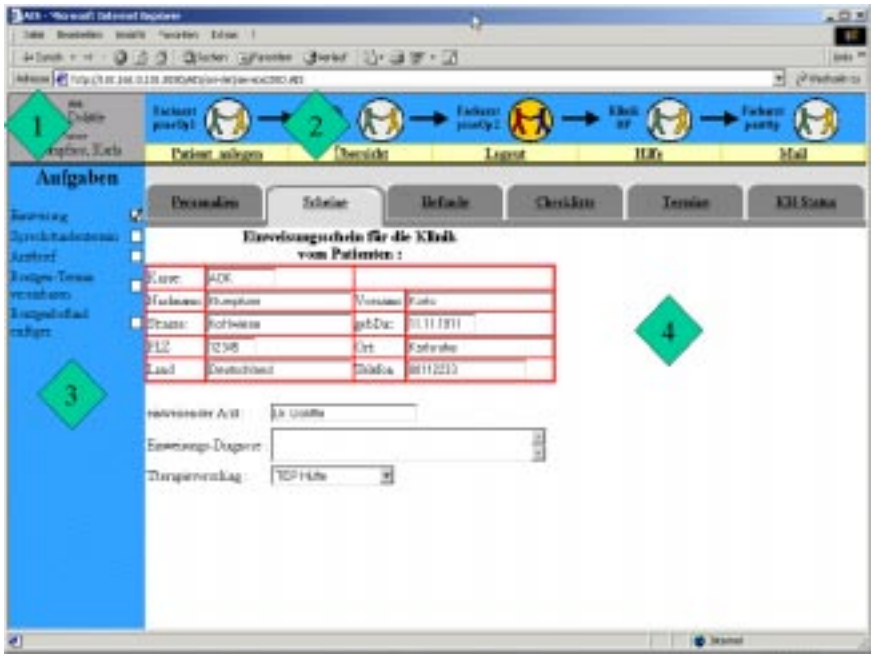

Figure 6: Support at the Servicepoint

A typical work scenario could be as follows. A specialist starts the system and chooses the serviceflow "Preparation, performance and aftercare of a stationary operation". Then, a window opens with his/her logo, the serviceflow in question, an empty task list and an overview over his/her patients being active in the serviceflow. Out of those the specialist chooses a patient in which he/she is interested.

As soon as this happens, the patient record appears. It includes general information about the patient, forms exchanged during the process, results of investigations and a checklist capturing the history of the process in detail together with every planned activity, extra accumulated appointments and information provided during the hospital stay, see figure 7. Furthermore and according to the point of service the patient is in, the physician might work on a certain service point. Here a task list with standard tasks at this point is delivered. While clicking on each of the tasks, the patient record will be opened at the right place to support the specialist's work. If the work is performed, the task will be marked automatically (or per hand). Even though possible in the underlying serviceflow representation and implementation, the interface is not yet supporting changes in the task list. Additionally, the interface lacks to provide a list of pre- and postconditions.

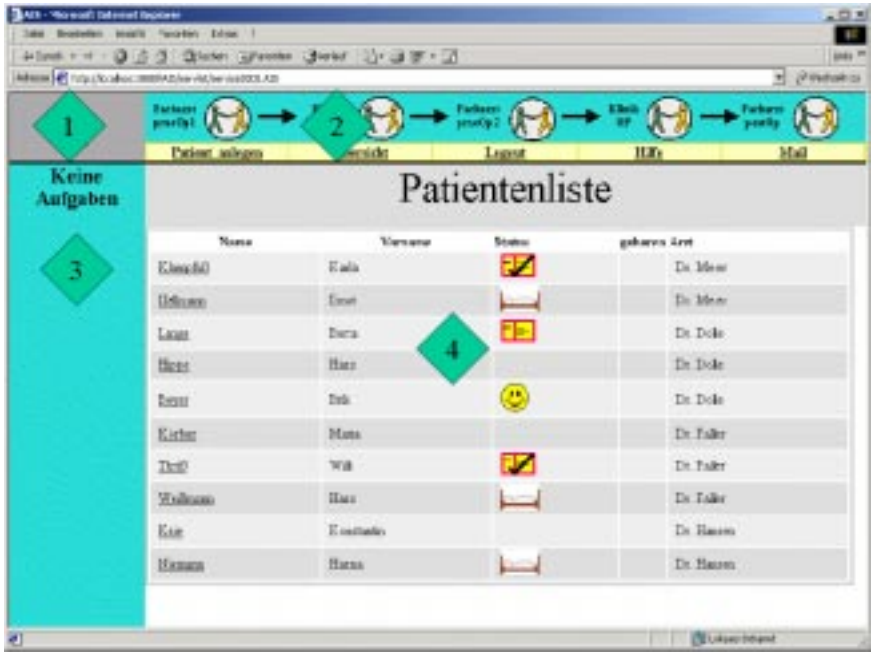

Figure 7: Overview over patients

As a deviation from the original SFM concept and due to the "one server solution", the prototype allows at each time to be informed about the patient's actual case (not only the copy of the sent servicefloat) as well as to add documents to the process which requires further indicators of changes (awareness information) at the overview and patient record level.

[18]

A.2 Case: the postal vote application at www.hamburg.de

A.2.1 Serviceflow model for postal vote

In our case of citizens applying for postal vote through the web portal of the city state of Hamburg (Germany) we have identified four service points with respective activities/operations in parentheses (see figure 8):

1. providing application assistance for citizens at the city's web portal www.hamburg.de (opening application, autooperational assistance in personalization, on-site evaluation, confirming reception, serviceflow preview, offering/registering personal reporting channel, optional: saving application)
2. inspecting the application at "Senatsamt für Bezirksangelegenheiten", the city's central administration for IT procedures (autooperational validity check including selecting the voting office in charge; or exception handling: selecting the voting office in charge if application processing seems possible – or moving directly to service point 4 in case of invalid application)
3. processing the application by the respective voting office (validity check with up to date preconditions, preparing personal postal vote ballot, notification for the electoral register, preparing postal vote ballot for dispatch, personalized exception handling if necessary)
4. reporting on process by the web portal provider (delivering messages to inform applicant about state of the process, providing information about what to do next and/or who to contact) through the channel the applicant has selected before (web page, email, SMS, etc.)

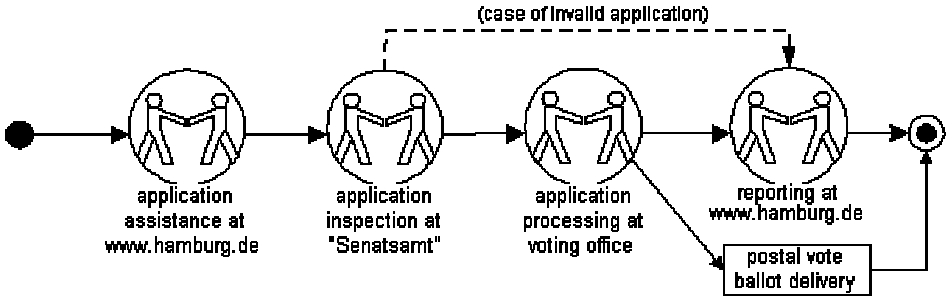

Fig. 8. Serviceflow model for the postal vote application starting at www.hamburg.de

Other activities/operations not focussing on or reflecting the citizen's personal/situated need are considered support processes, in this case the delivery of the postal vote ballot by regular mail.

A.2.3 Excerpt of a Servicefloat

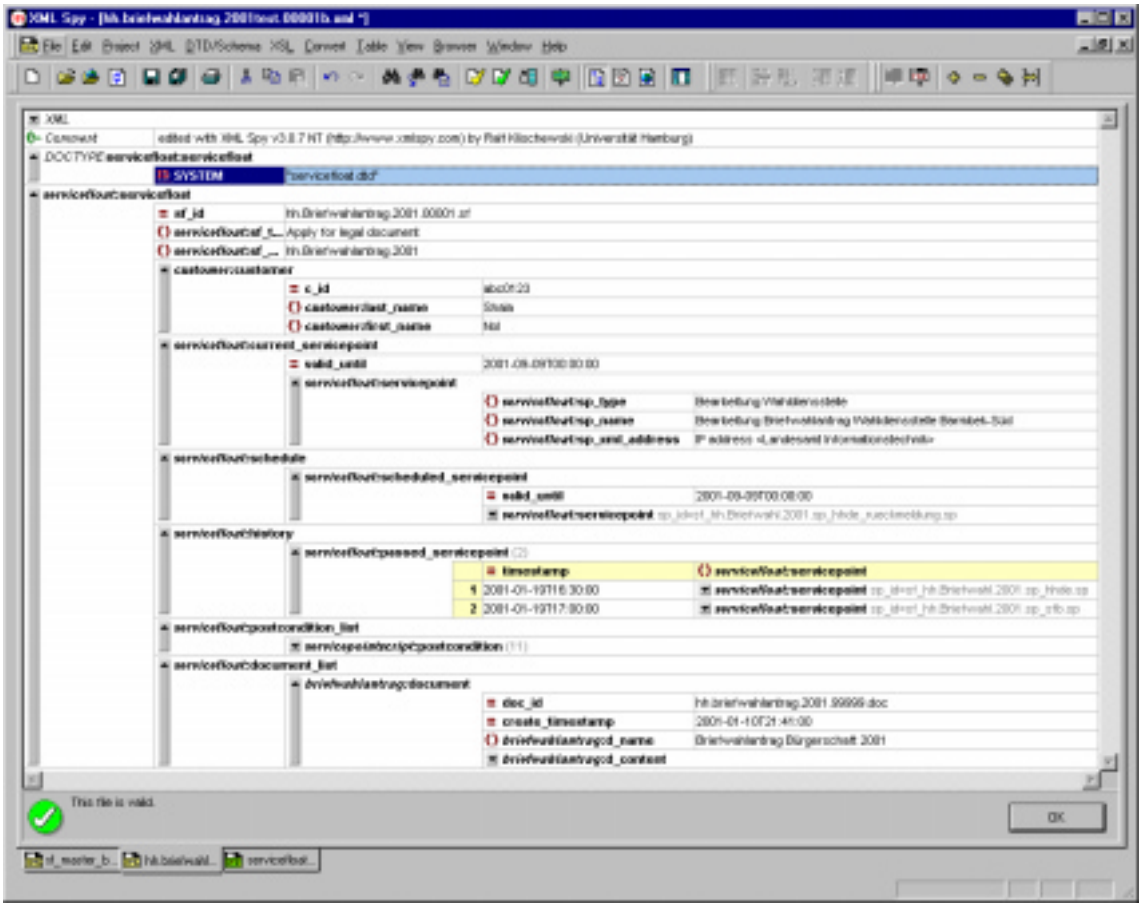

Figure 9 shows the XML representation presented with XML-Spy Editor.

Each servicefloat contains the following elements

- identifier for individual serviceflow (based on serviceflow type/variation)

- basic information on serviceflow client (with possible reference to comprehensive client data)
- current servicepoint (servicepoints are described by identifier, name, type, provider, address)
- list of scheduled servicepoints
- list of passed servicepoints
- list of accumulated postconditions
- list of documents, i.e. short message texts or references to full documents or document folders

The corresponding XML-representation is given in figure 9.

A.2.2 Course of the Project

While the process described above seems pretty straight forward (at least simple enough for prototyping purposes), a number of variations, uncertainties, possible exceptions and failures may occur. Situated needs to be addressed include that a citizen

- moves to a new address before the voting offices starts processing his/her application (voting offices open only a few weeks before the election date)
- has lost the postal vote ballot and needs a new one
- does not need to use postal vote and wants to vote at the polling station

However, the administration expects that the majority of the personalized serviceflows will follow the designed pattern. In our case, the above serviceflow model is the adopted basis for cooperation between the different service providers: the commercial portal provider of www.hamburg.de, the city's central department for application programming ("Senatsamt für Bezirksangelegenheiten") and the city's voting department responsible for the temporary voting offices. Also, the city's finance department, responsible for e-government strategy, is involved in the background. All parties involved have acknowledged that the underlying concept of serviceflow management applies a general perspective and that the case of applying for postal vote is only one first example to demonstrate the city's new capabilities and to learn how to manage the organizational and technical aspects of e-government transaction services.
At the time of writing (March 2001), all parties have (1) allocated personal/financial resources to implement IT support for realizing the respective service points and (2) agreed to a research partnership with the university's Department for Informatics (by which the authors can participate in, contribute to and evaluate the implementation process). The concept as well as a prototype for the portal's service point has been developed and presented at the university in January. In May, all service points should be ready for testing, and the service should be available in July to support the city state government election in September 2001.

[22]