SPACE: An Architecture for Coordinated Intra-European Assembly and Exchange of Citizen Data

Peter D. Holmes, Arve Larsen, Michael Gritzman
[peter.holmes@nr.no] Norsk Regnesentral, N-0314 Oslo, Norway

Stein Myrseth
New Media Science ASA, N-0304 Oslo, Norway

Abstract

Within the European Community, the Maastricht treaty grants European Citizens free movement between Member States. These citizens currently face a number of administrative barriers when planning and carrying out such moves, however.

The primary objectives of the SPACE Project have been to develop the foundation of an architectural framework for an operational SPACE system; and to create a prototype which demonstrates the possibility of providing a ***S****ingle* ***P****oint of* ***A****ccess for* ***C****itizens of* ***E****urope.*

The SPACE system design and its prototype currently demonstrate the provision of the kinds of administrative services required when moving between EU Member States. The design is based upon a telematic infrastructure for the retrieval, assembly and international exchange of Moving Advice and Citizen Data. In addition, inspection of SPACE's architectural framework reveals that its design can be applied to problems within other domains.

1 Introduction

1.1 Background

The SPACE Project has received about half of its funding from the EU's Telematics Applications Programme (Project AD 1014). The total Project budget was 7508900 ECU, and the work has been carried out over a 2 ½ year period (January '96 through June '98). The Consortium Partners have been CSC Danmark (Coordinating Partner, Denmark), debis Systemhaus (Germany), Cap Gemini (Netherlands), ICL (UK), INDRA (Spain), Sema Group (France), SOGEI (Italy), Tieto Group (Finland) and Norsk Regnesentral (Norway).

1.2 Problem Statement

The Maastricht treaty grants European Citizens free movement between Member States within the European Community. However, these citizens currently face a number of administrative barriers when planning and carrying out such moves. These barriers include:

1. ascertaining precisely which rules and regulations are applicable to their own, particular moving situation; and,

2. acquiring the certified documents which will be required in order to register (and de-register) themselves with a variety of Administrative bodies/sectors in the destination (and departure) countries. In general, such documents contain information about the Citizen which is stored within electronic archives controlled by such various Administrations.

In order to receive required information, deliver or certify documents, etc., the conditions above render it necessary for Citizens to contact numerous public bodies — and sometimes visit them more than once.

1.3 Goals

One of the objectives of the SPACE Project has been to demonstrate the possibility of providing a Single Point of Access for administrative services related to moving within the EU, based upon a telematic infrastructure for the retrieval, assembly and international exchange of such information.

Using the SPACE system, authorized civil servants can help provide Citizens with one-stop shopping of administrative services. Using data which is more accurate and reliable, both Citizens and Administrations can benefit from a more efficient registration process.

The goals of the SPACE project have been:

- to develop the system concept and the SPACE system's basic, architectural framework;
- to develop a Demonstrator based on state-of-the-art technology, in order to illustrate the concept and its

benefits, as well as help identify the functionality required for an operational system; and,

- to present the concept and the Demonstrator for various interest groups, in order to verify the functional and technical concepts, as well as obtain willingness to support the project towards an operational implementation.

1.4 Design Conditions and Openness

An operational SPACE system must connect and interwork with existing **citizen data systems** (CDSs). Citizen data systems are systems from which SPACE-relevant citizen data is to be retrieved. In the real world, these citizen data systems are of heterogeneous types. Therefore, the SPACE system design has been made open so as to cater for connection to and communication with heterogeneous CDS environments, without demanding changes to the existing systems. This condition is mandatory in order to protect existing investments in Administrations' technical infrastructure.

Within an operational context, the design of the SPACE system must enable the possibility for requesting and transmitting citizen data across different Member States. The SPACE system design includes such functionality, in order that the Demonstrator can illustrate the SPACE vision. In certain States, however, it is not yet politically and/or legally acceptable to perform such actions: some States may wish to restrict access and delivery of certain kinds Citizen data. SPACE's architectural design also includes an approach by which to control access and release of fine-grained citizen data elements. SPACE has the capacity to retrieve, assemble and exchange citizen data across European borders, and this functionality can be tailored to meet both the existing and future political / legal frameworks within the EU.

More and more of today's technology is based upon on-line communication. Still, a good number existing CDSs are currently based upon message-based communication (e.g., X.400, EDIFACT). Therefore, the SPACE system design accommodates for these kinds of off-line communication connections, as well.

Certain organizations may find it preferable to employ different kinds of work styles and organizational procedures when supplying Single Point of Access services to Citizens. The SPACE system's architecture is therefore open to accommodate various kinds of work patterns.

1.5 SPACE's Approach: Advantages and Achievements

The SPACE system's technical foundation is based upon a distributed, object-oriented architecture. SPACE chose CORBA [1] as the technology for addressing the problem of technically integrating a heterogeneous collection of CDS infrastructures. A system such as SPACE, which could potentially transfer sensitive information between EU member states, emphasizes the requirement for prioritizing security on several levels and types. Through the use of CORBA, SPACE enables a more uniform approach to handling security. As a basis technology, CORBA also enables the development of well-organized and controlled system management routines.

Using a distributed system design, it is sufficient that each data item and information unit have a single instantiation somewhere in the information space: they *need* not be replicated anywhere. Mirroring *can* be used, however, in order to increase performance and efficiency. Use of a single-source approach for data and information maintenance helps in the effort to eliminate inconsistent information.

SPACE has achieved the creation of object interfaces and service mechanisms which are highly generic in nature. This achievement makes SPACE a highly extensible system, enabling the straight-forward addition of new countries and administrative sectors.

In the Demonstrator, the SPACE Client is designed and implemented as a JAVA applet. This enables platform independence, as well as eliminating the need to install, maintain and update SPACE Client software from machine to machine. SPACE has also achieved the design and development of a multi-lingual user interface.

One of the important aspects of the SPACE system design is the manner in which it exploits the combination of CORBA and JAVA. This capacity is particularly advanced within the Dialogue Mechanism — a mechanism which operates within the SPACE Client's interactive user interface.

SPACE has the capacity to access information found within a diverse and independently organized collection of European Administrations. Seen from this perspective, the SPACE system achieves *enterprise integration*. Inspection of SPACE's architectural framework further reveals that its design can be applied to problems within other domains. Common to these problems is the natural division of each high-level service request into a set of more specialized service requests, followed by delegation of these specialized requests to a set of subservient objects.

1.6 Structure of the Paper

The remainder of this paper describes the *functional scope* of the SPACE Demonstrator and the Information Products delivered by SPACE. Thereafter, aspects of SPACE's distributed system architecture are presented, followed by a brief description of the current organization of the Demonstrator. The characteristics of the user interface are then highlighted, followed by a section describing the conditions upon which the SPACE architectural framework can be reused within other domains. The paper closes with conclusions, status and remarks about future plans.

2. Functional Scope of the SPACE Demonstrator

The purpose of this section is to provide the reader with a framework for understanding the scope of the work carried out in regard to the design and implementation of the SPACE Demonstrator.

2.1 The Administrative Sectors

The SPACE Demonstrator addresses information access and delivery across five different sectors: Tax, Civil Registration, Vehicle Registration and Drivers License, Health Care and Social Security. SPACE has assessed requirements from these sectors across each of nine different Member States.

2.2 Principal Use Scenarios

One of the results of the User Requirements Analysis phase of the project has been the identification of high-level business processes which were essentially common to all of the sectors addressed by SPACE. Using one view of these business processes, it is possible to characterize them as scenarios for SPACE system use. Short descriptions of these principal use scenarios are summarized below.

Scenario 1: Ask for Advice: A citizen, who has an intention to migrate, may approach a SPACE office and seek advice and guidance on the processes, procedures and information requirements of his proposed destination Member State. In addition he/she will get advice about what to do in the originating country prior to departure.

For example a citizen from country X wishes to move to country Y to seek employment. Before he goes he wishes to know exactly what he will have to do when he arrives in country Y: where he will have to go and what documentation he will need.

Scenario 2: Build Portfolio / Migration Preparation: An EU citizen has decided to migrate from country X to country Y and asks SPACE to assemble a package of personal information and data — assembled together into a "Portfolio" — which will be required by the destination administrations.

Scenario 3: Make Portfolio Available: This scenario implies movement/transfer of the data package (i.e., Portfolio) to the destination member state and making it available in a usable format to the administrations. This includes addressing all provisions necessary for authentication and validation, together with other security aspects. This scenario also implies addressing the processes which will have to be put in place for handling the data package when it arrives in the destination state. This could range from:

- informing the administrations that such a package exists and making relevant parts of it available for read access to each administration, under the access rights authorized by the originating administrations;
- display of the information within the SPACE Client's user interface;
- print out for the citizen; etc.

2.3 Demonstrator Functionality

An assessment of the functional requirements for the SPACE System followed the User Requirements Analysis phase of the project. During this effort, it was judged necessary to partition the requirements.

One partition consisted of the functional requirements deemed to be an essential part of the SPACE business kernel. The more complex and specialized functional requirements — requirements demanded of an operational system, as well as those unique to specific sectors and/or countries — were separated from the kernel set.

Of the kernel set, efforts have been made to focus upon the implementation and presentation of system elements required for a convincing demonstration of the SPACE concept. The primary functionality of the SPACE Demonstrator therefore includes:

- a multilinguistic user-interface
- the capacity to specify and input details about a Citizen's particular moving situation
- the capacity to generate both General and Tailored Advice
- the capacity to browse and print Advice
- the capacity to determine which kinds of data (*data fields*) will be required for various registration processes within the Destination State (and de-registration processes within the Departure State)

- the capacity to determine which of those data fields can be populated using information found within CDSs
- the capacity to determine which keys are required from the user in order to retrieve the information existing within those CDSs (e.g., if the Destination State requests the Citizen's "Wedding Date", the Citizen's person-number may be required in order to retrieve the value of "Wedding Date" from the Departure State's Civil Registration CDS)
- the capacity for the Citizen to specify values for keys
- the capacity to build a Portfolio containing automatically-instantiated data fields — particularly those fields required for various registration and de-registration processes
- the capacity to manually specify a value for such a data field, e.g., in cases where no CDS is able to supply a value for that field (note: this function serves to enable the completion of a partially-instantiated Portfolio)
- the capacity to browse and print Portfolios
- the capacity to transfer Portfolios between Member States.

3. Information Products Delivered by SPACE

3.1 Advice Packages

Advice Packages contain the kind of information one might normally publish and have available as *brochures* at some administrative office. In the SPACE system and for the end-user, Advice Packages can contain General Advice and/or Tailored Advice.

General Advice is information which can be provided about moving within the EU when only knowing things such as the Citizen's planned departure and destination states. Tailored Advice, on the other hand, is information which is especially tailored to fit the Citizen's moving situation. Customizing advice in this manner requires greater knowledge about the details of the moving case. Such details are gathered from the Citizen via an interactive dialogue process which runs on the Client.

As part of the General and Tailored Advice SPACE can deliver, SPACE can also present facsimiles of actual registration forms from various countries, when available.

3.2 Portfolios

SPACE Portfolios are created during the "Build Portfolio / Migration Preparation" Business Process. The major elements of a Portfolio are the Citizen Input Data Package and the Sector Data Packages; these latter Packages contain the citizen data retrieved from CDSs. Portfolios can be delivered to the end-user, as well as transferred between EU countries.

4. SPACE's Distributed System Architecture

4.1 The Basic Object Types

The SPACE system is based upon five major component types: **SPACE Clients**, **SPACE Masters**, **SPACE Experts**, **CDS Objects** and ***real CDSs*** (Citizen Data Systems). Real CDSs are the actual systems from which SPACE-relevant citizen data is retrieved. Of these five components, three are "true" object types: the SPACE Master, the SPACE Expert, and the CDS Object. IDL is used to define the service interface for each of the Master, Expert and CDS Objects.

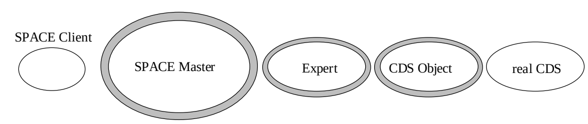

Figure 1 Basic system components

The SPACE Client is the part of the SPACE system that runs on the desktop computer and, in fact, is the end-user's (i.e., SPACE Clerk's) access point to the SPACE system. Generally speaking, the SPACE system is driven by events generated via the SPACE Client's user interface.

Events generated at the SPACE Client invoke services offered by a SPACE Master. SPACE Masters interact with SPACE Experts which, in turn, interact with CDS Objects. CDS Objects encapsulate the access, retrieval and transformation of all data retrieved from the real CDSs.

In an operational setting, there will exist one or more instances of a Master object per country (e.g., Norwegian Master(s), Danish Master(s), etc.). There will also exist one or more instances of an Expert object per country per sector (e.g., Finnish Civil Registration Expert(s), French Social Security Expert(s), etc.). Expert objects interact with instances of CDS Objects, which may optionally be subtyped. Each instance of a CDS Object (sub)type is associated with exactly one real CDS.

4.2 Generic Design Approach

The service interfaces for the SPACE Master, the Expert and the CDS Object are designed in a highly generic fashion. Associated with these generic services are a collection of generic DB table structures; these include the Master Routing Table structure, the Expert Routing Table structure, the CDS Key Table structure and more. The role of the generic DB table structures is to support the generic services.

The SPACE design is such that it is possible to use *one* software executable (e.g., `Master.exe`) for all instances of Master objects *across all countries*. When creating instances of Master objects for different countries, different sets of command line arguments are used. The arguments serve to inform each object instance as to which *table* instance (e.g., which country-specific instance of the Master Routing Table) the object should refer to when retrieving information (e.g., request routing information). These same design and instantiation conditions also hold true for the Expert object type and – except for one service – the CDS Object type as well.

The goal of the design and implementation work for every generic service has been to create the services such that they can be reused by all countries and/or sectors. For this reson, the implementation of the Master and Expert service interfaces are completely free from policy-related mechanisms. Reuse of the SPACE Demonstrator's basic object types essentially amounts to employing different instances of Master Routing Tables, Expert Routing Tables, etc., and populating those instances with different data. It is only in the CDS Objects that a single service (i.e., the `get_cds_data` service) must be implemented uniquely. This service cannot be made generic due to the fact that different, real CDSs will store Citizen data in different manners (e.g., using different DB tables structures, etc.), thereby making it necessary to create CDS-specific data access routines.

4.3 Real CDSs

Each real CDS is best conceived of as a single logical entity. As a logical entity, a real CDS could be technically implemented as a single database server. Alternatively, another such logical entity could be technically implemented as a set of geographically-distributed, interoperating systems. *It is also possible that what are logically unique, real CDSs may share machine / network / database resources; in fact, the same piece of data may belong to more than one logically unique, real CDS.* Examples of real CDSs, as logical entities, are depicted in Figure 2.

From a conceptual perspective, each CDS Object *type* is associated with exactly one logically unique, real CDS (and vice-versa); at execution time, however, multiple *instances* of that CDS Object type may be associated with that real CDS.

There are very good reasons for having a one-to-one correspondence between CDS Object types and logically unique, real CDSs. Having such a correspondence allows for the creation of logically distinct *data channels*. In SPACE, these data channels are oriented according to sector. Since the same piece of data may belong to more than one logically unique, real CDS, that same piece of data may be allowed to pass through more than one data channel.

One of the responsibilities of the CDS Object is to ensure that CDS data is only released to Administrations authorized to have access to that data (e.g., country X may presently allow the value of "Wedding Date" to be released to country Y's Civil Registration authorities, but not country Y's Tax authorities). The CDS Object is therefore the point at which localization of *filtering policies* are best implemented, see [2]. Use of logically distinct data channels simplifies the manner in which such policies can be implemented.

Creation of logically distinct data channels also eases matters from a business perspective. That is, the definition and implementation of each CDS Object type is the "point of contact" between the SPACE System and the systems which can supply citizen data. This condition implies that CDS Object types play a central role when it comes to contractual, business and technical negotiations between SPACE and the Administrations "owning" and controlling citizen data. Establishing and maintaining business contracts which involve access and delivery of data within a single data channel is far more simple than doing so when data channels are mixed (i.e., mixing the access and delivery of cross-sectoral data within a single channel would require simultaneous negotiations with multiple partners).

Figure 2 illustrates the manner in which logically distinct data channels are created through the one-to-one correspondence between CDS Object types and logically unique, real CDSs. It is interesting to note in the figure the exemplification as to how more than one CDS Object type can be defined for the same sector (e.g., the CDS Object types: Soc. Sec **1** and Soc. Sec **2**). These two different CDS Object types are required since there have been defined two logically unique real CDSs which can deliver Social Security information.

4.4 Required support elements

The design of the SPACE System is such each of the system's basic object types requires certain technical elements which support its functionality. SPACE Master objects and SPACE Experts objects require WWW Server support and a database system. CDS Objects require a database system. The SPACE Client requires a

WWW Browser. Clarifications of the roles of these support elements are found in [2].

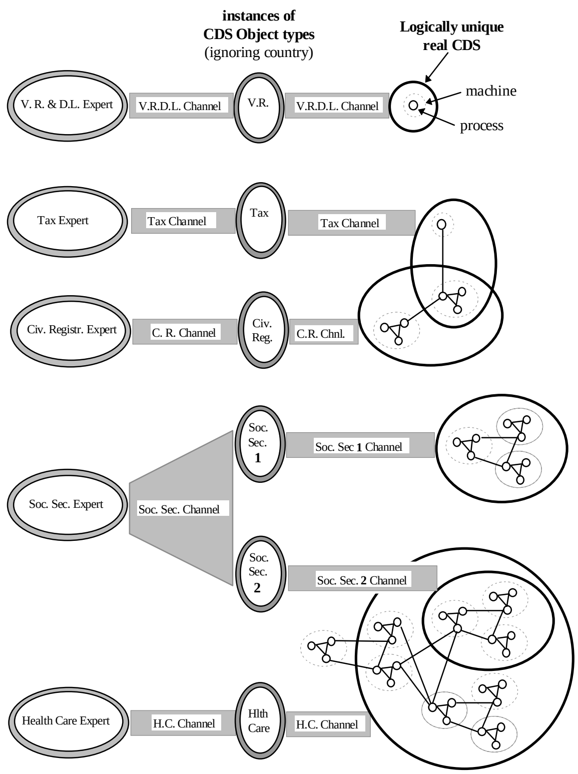

Figure 2 : Logically distinct data channels

5. The SPACE Demonstrator

The SPACE Demonstrator, as a prototype, is *not* connected to any factual CDS systems running in a production environment. Instead, data sets have been created which liken the structure and content of the data sets found in CDS production environments. In the SPACE Demonstrator, these data sets have been stored within what are called **PDSs** (Pilot Data Systems), in contrast to CDSs.

Figure 3 depicts the technical configuration of the SPACE Demonstrator. In this figure, the technical relationships between only two machines is indicated; when the Demonstrator employs more than two machines, the technical relationships are analogous to those already depicted.

Figure 3 illustrates that each SPACE Server machine includes a WWW Server. On each of the WWW Servers' file systems is a copy of SPACE's JAVA-based Client. In addition, these WWW Servers contain WWW pages relevant to certain services delivered by SPACE (e.g., delivery of electronic facsimiles of registration forms, etc.).

The SPACE Client employs two logical connections to the SPACE Server machine: one is an HTTP connection to the WWW Server on that machine, while the other is a IIOP connection to the SPACE Master via the ORB. The functional interdependence between the SPACE Master and the WWW Server is explained in [2]. The figure also depicts that a SPACE Client can access WWW Servers found on more than SPACE Server machine; in fact, SPACE Clients can (theoretically) access WWW Servers on machines outside of the SPACE environment. This open aspect of design allows for information delivered by SPACE to include WWW links to useful information stored and maintained outside of the SPACE environment. Whether and how this design feature should be employed by SPACE and/or the User Organizations is a matter for further study.

Figure 3 illustrates that the Expert object for Civil Registration has connections to two different CDS Objects. This indicates that, in addition to PDSs, the SPACE Demonstrator has been connected to a "mirror" of the production system used for the Civil Registration sector in Finland.

6. SPACE Client User Interface

6.1 Design Philosophy

For the SPACE Demonstrator, the SPACE Client user-interface design is targeted for operation by a trained clerk. It is targeted for a user who, at least on a semi-regular basis, operates the system. *The SPACE Client user-interface for the SPACE Demonstrator is not aimed at users wholly unfamiliar with the system and its purpose.* An interface design for unfamiliar users (e.g., an interface designed for public access via Internet) would be based upon philosophical design principles similar to those presently used, but it's "look and feel" would be quite different than that found in the Demonstrator.

The user-interface design is derived from the functionality which the system should provide. The design philosophy employed is based upon a *task-oriented approach* [4]. Using a task-oriented approach as a guiding principle for user-interface design implies that the design which results does not adhere any of the defacto standards (e.g., Motif, Windows, etc.).

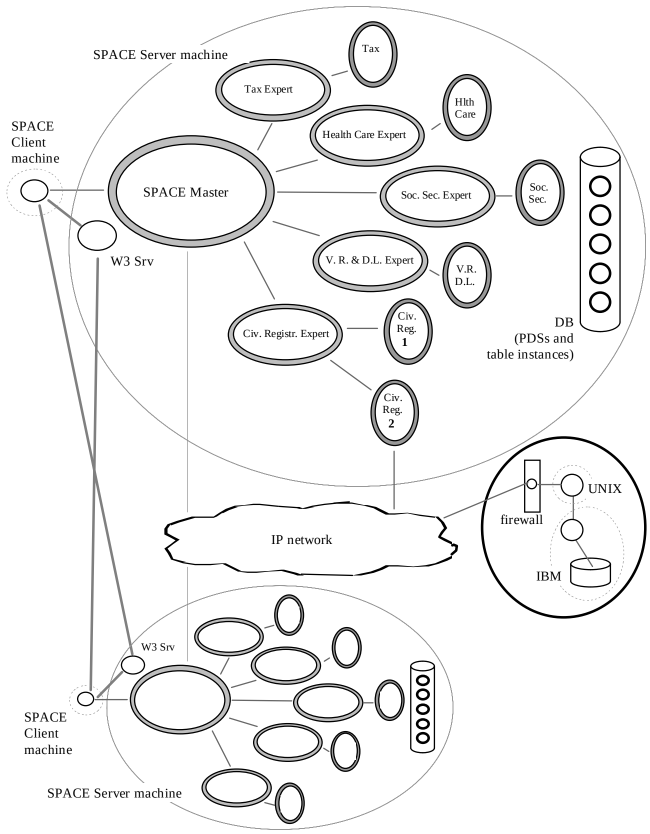

Figure 3 The SPACE Demonstrator

6.2 Design Principles

The overall philosophy of the user-interface design is reflected in the following set of design principles:

- Single screen-page approach. The user-interface consists of only one screen page. All system functionality is expressed on that screen page. The single-page approach emphasizes the system as a special-purpose system, where functionality is strictly limited to handle the functions required within the defined use-scenarios. This approach eliminates the need for navigation mechanisms between different screen pages. All functionality offered on the page is directly connected to the core functionality.
- "What-you-see-is-what-you-can-do-right-now". Sequencing is an important notion in relation to the functionality offered by the system. Input from the user and user other actions are required in well-defined sequences. These sequences can be grouped into "action sequence envelopes". Such an envelope does not define a single, strict sequence of interface operations, but rather a set of inter-related sequences. The user-interface accommodates this approach by means of a design principle called "What-you-see-is-what-you-can-do-right-now" [3]. For example, at any point during the user's interaction with the system, only relevant actions are offered to the user (i.e., actions which can/should be taken at that given point). Actions which are not relevant to the current task workflow are simply not offered.
- Elimination of dedicated sequencing mechanisms. A consequence of the two points above is the elimination of the need for *dedicated* sequencing mechanisms within the user-interface (e.g., "go to next page", "select one of the following"). Having achieved this, there is no need for user interface artifacts such as dialogue boxes, menus, etc.
- Reusable user-interface design. Despite the fact that the user-interface is designed in a task-oriented manner, the principles underlying the interface are still very much reusable. Any future implementation of the SPACE System — which may well contain more countries, more sectors and more functionality — can still utilize these basic design principles.

6.3 Multilingual User-Interface

The SPACE design includes a flexible and extensible approach for achieving multilingualism within the Client user interface. The approach for achieving multilinguistic renderings of all textual information within the system is as follows:

Each text-related element of information which can appear on the screen is assigned a unique, language-independent identifier. For each such identifier, language-dependent renderings of the text associated with that identifier are created and stored in a database. To render any textual element, it is only necessary to retrieve the desired rendering from the database. Here, two keys are sufficient for database access: the text-element identifier and the desired language.

It is never the case that data retrieved from CDSs is translated. At most, syntactic transformations upon CDS data may be performed (e.g., changing the format for dates, etc.). Any and all syntactic transformations of this kind are performed by CDS Objects, as part of the their encapsulation of real CDSs.

6.4 Dialogue Mechanism

One of the important aspects of the SPACE system design is the manner in which it exploits the combination of CORBA and JAVA. This capacity is particularly advanced within the Dialogue Mechanism — a mechanism which operates within the SPACE Client's interactive user interface. This mechanism employs SPACE's Dialogue Definition Language (SDDL), which enables the flexible specification and encoding of user dialogues. More generally, this mechanism is an architectural instance of Unified Dialogue within a Distributed Object System, as described in [5].

The Dialogue Mechanism works in the following manner: during operation in the "Get Advice" and "Build Portfolio" modes, dialogue specifications are passed from the SPACE Masters (in both the Departure and Destination States) to the SPACE Client. There, the Client interprets those specifications — a process which engenders an interactive dialogue with the user.

Once the questions originating from the Masters have been answered, the replies are passed back to the sector Experts in each of the Departure and Destination States. In the Experts, the replies are analyzed. Should any sector Expert wish to pose further questions, it simply passes a dialogue specification back to the Client via its associated Master. This organization of dialogue specifications and the manner in which they are interpreted in the SPACE Client achieves two extremely important goals:

1. high-level, general questions can be asked first and, depending upon the replies submitted, more detailed and *relevant* questions can be asked if necessary; this helps contribute to the user's experience of a less tedious, more intelligent and friendlier system; and,

2. the interactive dialogue to be carried out with the user is *not* hard-coded into the SPACE Client, thereby allowing countries and sectors to dynamically update their individual dialogue specifications when necessary; this helps achieve greater independence of development across countries and sectors.

Further material describing an architecture for Unified Dialogue within Distributed Object Systems can be found in [5]. A discussion concerning the relationship of SPACE's requirements to the grammatical features in SDDL can be found in [2].

7. SPACE: A Reusable Architectural Framework

Inspection of SPACE's architectural framework reveals that its design can be applied to problems within other domains. The current incarnation of the SPACE architectural framework and its generic design can be typified as one which effectively accomplishes *service request delegation and reply assembly*. As SPACE is presently defined, these two kinds of processes are found at three levels: between Masters and Masters, between Masters and Experts, and between Experts and CDS Objects. In addition, SPACE's delegation and assembly processes have the form of a tree; there is currently no delegation and assembly interaction across subtrees. As suggested in Figure 4, this kind of service delegation structure could by extended ad infinitum.

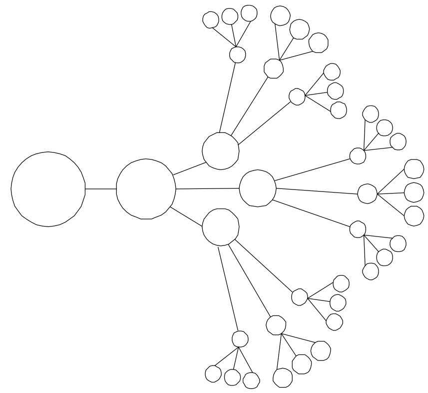

Figure 4 Service request delegation structure

In SPACE, assembly of return values from a set of subservient objects (all of the same type) is essentially one of two kinds: assembly by performing a union operation upon the return values, or assembly by creation of an unordered set containing the return values. This condition makes it possible to devise and employ reusable patterns for implementation of assembly routines.

Furthermore, this tree-based service delegation and reply assembly approach yields a dynamic infrastructure wherein nodes and levels can be added and removed within the infrastructure during runtime — without requiring notification to all nodes in the structure.

The service delegation structure and generic service mechanisms found in SPACE are quite straight-forward, perhaps even overly simplistic. However, it is this same simplicity which yields the extensibility and country/sector independence required within the problem domain addressed by SPACE.

Highlights and Conclusions

- Through the information products it creates, SPACE seriously addresses the problems Citizens and Administrations face when Citizens move within the European Union.
- SPACE has designed an open and extensible object-oriented architecture, based upon a *generic design approach.*
- SPACE's distributed architecture enables:
  a) distribution, replication and redundancy of services within an international information network

b) the capacity to deploy policies for information access and release in a country- and sector-specific manner, and
c) the development of a common intra-system model for addressing security.

- SPACE's use of JAVA eliminates installation and local maintenance of SPACE Client software within the User Community.
- SPACE has developed a robust approach by which to address the problem of multilingualism.
- SPACE achieves *enterprise integration* through its capacity to access and assemble information found within a diverse and independently organized collection of European Administrations.

9. Status and Future Work

At the time of writing, the SPACE Demonstrator has been constructed, and SPACE Demonstrations are being carried out in a number of European countries. The purpose of these Demonstrations is to validate the SPACE System Concept, to receive feedback concerning the system's functionality and to promote interest across a variety of potential user groups and Administrations. In addition, the Consortium is developing a multi-model exploitation plan by which to achieve further continuance and deployment. In particular, marketing and exploitation activities are being especially focused upon the development of a Nordic pilot for SPACE, as well as relations to the ongoing IDA work, see [6].

References

[1] The Common Object Request Broker: Architecture and Specification, Revision 2.1, August 1997. *http://www.omg.org/library/specindx.htm*

[2] Holmes, P., Larsen, A., Gritzman, M., Lundsgaard, L., Skardhamar, H. , Pohjosmaki, R., Khanh, T.-H., "SPACE client and server specification: Infrastructure specification for the SPACE technical platform". EU Project Report: SPACE Deliverable D902/D903 (Confidential). Also reprinted as NR Technical Note Nr. IMEDIA/06/97, December 30, 1997.

[3] Gritzman, M., Kluge, A., Lovett, H., "Task orientation in user interface design", in *Human-Computer Interaction*, Interact '95 (Eds. Nordby, K., Helmersen, P., Gilmore, D.J., Arnesen, S.A.), Chapman and Hall, London, Glasgow (1995), pp. 97-102.

[4] Norman, D.A., "Why Interfaces Don't Work", in Brenda Laurel (ed.) *The Art of Human-Computer Interface Design*, Addison-Wesley Publishing Company (1990), pp. 209-219.

[5] Larsen, A., Holmes, P., "An Architecture for Unified Dialogue in Distributed Object Systems", *To appear in* the Proceedings of the TOOLS USA '98 Conference, 3-7 August, 1998, Santa Barbara, CA. *Also availale as* NR Technical Note IMEDIA/03/98, Norwegian Computing Center (NR), 1998.

[6] Architecture of the Trans-European Telematic Networks between Administrations, IDA Report, version 1.1, 26 September 1995.