Transforming Formal and Informal Work Processes

Pedro Antunes, Nuno Guimaraes
IST/INESC - Elect. Engin. and Comp. Dep.,
Technical University of Lisboa.
Av. Rovisco Pais
1000 Lisboa, Portugal

Abstract

This paper emerges from the identification of problems in the computational integration of formal and informal work processes in organizations. We develop a common view of both processes, concerning in particular the aspect of control, and propose a mechanism for seamlessly transferring control and supporting transformations between processes. Transformations allow to re-engineer work processes. The transfer mechanism operates by identifying and switching appropriate transfer points, contexts and actors. From this approach results an organizational system where different but interrelated work patterns coexist.

Keywords

Organizational processes; workflow; group decision support systems; work modes

1 Introduction

This paper has its roots on a European project named ORCHESTRA, finished in 1996. ORCHESTRA was intended to define and develop a platform for the integration of several technologies in organizational settings, notably workflow, messaging and group interaction and decision support systems. A complete description of ORCHESTRA can be found in [8].

The core component of ORCHESTRA was a commercial groupware product named ELENIX MULTIMEDIA. ELENIX was, at the beginning of the project, an integrated client/server office product for MS-WINDOWS, WINDOWS NT and UNIX. The set of services provided in this package includes: multimedia document archiving (global, local, disconnected operation and dead archive, including the possibility of using optical storage), electronic mail and fax (including parallel and serial messaging), correspondence registration and other additional services, like agenda and addresses databases. This base architecture and functionality is illustrated in Figure 1.

According to the objectives of the project, ELENIX was extended in multiple directions. The new product[1] provides a set of services intended to shift its functionality from the restricted office environment into the global organization:

- *Workflow functionality.* Based on the mail/fax service, a workflow engine allows the implementation of organizational procedures.
- *Security features.* Applied to the document archive, they provide user authentication, document integrity, concurrency and version control.
- *Mainframe integration.* Supporting access to mainframe databases, using heterogeneous transport protocols, preserves backwards compatibility while new computational support is installed in the organization.
- *Enhanced messaging.* Intended to identify context of messages arriving to the organization. This service converts fax messages to text format (OCR server), analyses converted faxes and mail messages, in order to select departmental mailboxes, and automatically delivers messages to appropriate recipients.

Two other relevant services were developed and integrated with ELENIX during the project but are not part of the commercial product: the

[1] Currently marketed as ELENIX MULTIMEDIA 3.0 by SMD Informatica.

Third International Workshop on Groupware, CRIWG '97. Madrid, Spain, October, 1997.

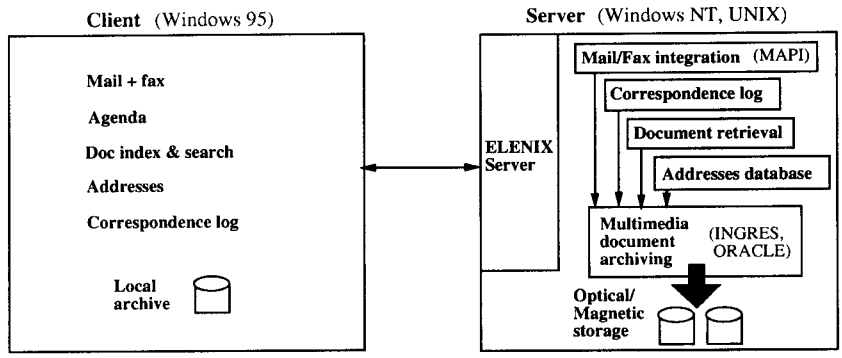

Figure 1: The ORCHESTRA core component

WF-EIS plug, which is dedicated to share data between workflow and an Executive Information System (EIS), by defining a common application interface; and the *group interaction and decision toolbox* [2], which groups a set of tools dedicated to support informal group decision processes (e.g. brainstorming). The later set of tools is usually coined as a Group Decision Support System (GDSS).

The architecture of the extended ORCHESTRA platform, including in particular the workflow functionality, was experimented in two pilot organizations: (1) the administrative department of a Portuguese thermal power plant[2] and (2) one Spanish organization charged with the management and acquisition of nuclear fuel for nuclear powerstations[3].

This paper departs from an analysis of the developed ORCHESTRA's components, specifically the workflow engine and GDSS, in order to discuss the integrated computational support for formal and informal work processes. A common view of both processes is developed, concerning in particular the aspect of control. Then, a mechanism for transferring control is proposed and, finally, transformations of formal and informal processes are defined.

2 Work and Control Patterns

From its beginning, the integration of the ORCHESTRA platform in organizations was conceptualized in light of Mintzberg's perspective on organizational structures [6, 7]. This perspective acknowledges the simultaneous presence of different structures and flows within organizations. One such structure is hierarchical, defining precise degrees of responsibility such that members of the organization as, for instance, middle line managers, exercise vertical control over members with lower responsibility, such as core operators. The hierarchical structure is complemented with a regulated structure, defining relationships and interactions among actors and groups of individuals with interdependent tasks. Another structure, designated work constellation, is intended to model more flexible, decentralized and un-regulated work processes. Work constellations congregate people with different responsibilities and knowledge, in groups which are mainly decision-oriented.

Workflow systems rely on formal (hierarchical and regulated) processes in order to define sequences of interdependent tasks, and identify specific actors and groups to perform specific tasks at specific times. When a given workflow procedure is enacted, work control is removed from users and delegated to the workflow engine, which guarantees that the process evolves according to the specified sequence. Usually, progress on a procedure, i.e. accomplishment of a task and request to perform the following one, is accomplished by transferring documents between users.

GDSS, which support informal (decentralized and un-regulated) processes, operate by defining phases in group decision-making, i.e. required steps to progress from the identification of a problem to the design and implementation of a solution [5]. Contrary to what happens with workflow enactment, the enactment of an informal process in a GDSS does not result in a specific instantiation of a procedure, with a specific

[2] E.D.P., Electricity of Portugal, one of the largest Portuguese companies.

[3] CONUC, a small organization of engineers and physicists.

sequence of decision-making tasks, since these activities are commonly characterized by unpredictable moves, as forward-backward progresses (for instance, to re-analyze problems) or recurrent progresses (e.g. identifying new problems). Nevertheless, informal process descriptions establish a decision-making strategy in a meeting, avoiding erratic progresses or other unwanted practices, as groupthink or choice shifts [4], while facilitating productivity gains.

Since informal processes should not be completely structured by GDSS, these tools organize the work process but delegate control to users. Users coordinate themselves by mutual adjustment. This functionality requires either to exchange messages between users or manage shared objects.

According to Mintzberg, both regulated and un-regulated modes of organizing and executing tasks are complementary and coexist in organizations. From a systems integration point of view, this means that both formal and informal work processes must be computationally supported and, furthermore, their corresponding control patterns must be preserved and seamlessly integrated.

2.1 ORCHESTRA's Approach

In ORCHESTRA, formal work processes were supported by the workflow engine, while informal processes were supported by the group interaction toolbox.

Formal processes were modeled using a commercial product named TASKON OORAM (Object Oriented Analysis and Modeling). The OORAM approach to modeling allows to construct different views of roles attributed to each object [9]: (1) area of concern, using descriptive text; (2) collaboration view, defining message paths between objects; (3) scenario, displaying time-ordered interaction sequences; and (4) process view, showing data flows along with actions performed by objects. Figure 2 shows a OORAM process view. Workflow modeling was strictly separated from description[4]. A workflow description tool and a workflow engine were developed by SMD Informatica.

[4] An approach recommended by the Workflow Management Coalition [10]. Although we acknowledge that mechanisms to automatically convert modeling data (conceptual level) to engine data (enactment level) are needed, this problem is not addressed in this paper.

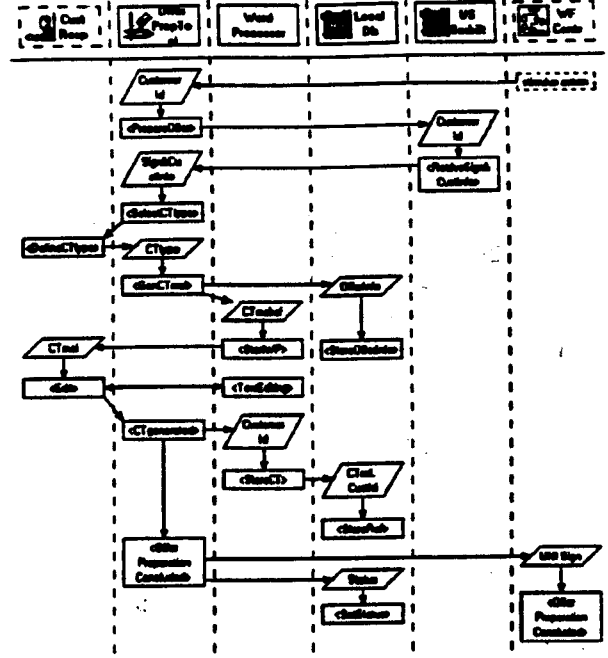

Figure 2: OORAM process view

Informal processes were modeled as sequences of independent and recurrent phases in group decision processes. Three phases were considered: (1) generation of ideas; (2) group discussion of ideas; and (3) prioritization/voting of ideas. A tool, named NGTOOL [1], was developed to support this model.

2.2 Transfer Control Between Formal and Informal Processes

The ORCHESTRA system primarily supports formal processes, operating in the following way. The workflow engine is in control of work processes until an exception is raised. Exceptions are originated by limitations in workflow procedures, due to the lack of formalization inherent to many organizational tasks. The limitations in workflow procedures require informal actions to be taken and, though, control is transferred from the workflow engine to NGTOOL. A tool, named RECOMMENDATION TOOL, accomplishes this task. The RECOMMENDATION TOOL enacts different phase combinations according to the nature of the problems, solutions and required tasks. The selection is accomplished by asking a standard set of questions to the person responsible for the interrupted workflow, and matching the answers with models of decision-making processes [3].

The above transfer mechanism was tested using data from one of the pilot organizations. The test case considered a workflow procedure concerning leased vehicles and damage reports to insurance companies. An exception was raised due to excessive damage reports for a single vehicle and required deciding either to warn, charge or dismiss the driver. The decision, not supported by the workflow procedure, required a brainstorming session between several managers of the pilot organization.

2.3 Evaluation of the Approach

A seamless transfer of control between formal and informal processes was not accomplished by the ORCHESTRA project. Although the workflow engine and NGTOOL were integrated using the RECOMMENDATION TOOL, the mechanism presents several limitations:

- The RECOMMENDATION TOOL acts only upon the emergence of exceptions in the workflow. Furthermore, the transfer of control from the NGTOOL back to the workflow engine relies on exception removal (for instance, by giving more time for accomplishing a task, or assigning different persons to the task). This means that the possibility of deploying informal processes as valid and seamless tasks within workflow procedures is not considered.
- The RECOMMENDATION TOOL is not able to identify who should participate in the informal process.
- The enactment of workflow procedures as consequences of informal processes is not considered, which means that control patterns across different work modes are not possible.
- Finally, the re-engineering of work processes is not envisaged. Although formal work modes are adequate for many organizational processes, one should expect that current practices can be improved if people are able to transform formal into informal processes. Of course, when productivity diminishes, or people become uncomfortable with the way informal work progresses, they should be able to step back into formal work modes.

The perception of the above problems leads to the requirements for the integration and seamless transfer of control between the workflow engine and GDSS; and support for transforming formal and informal work processes.

3 Developing a New Mechanism for Transferring Control

3.1 Transfer Points

We start by identifying transfer points, i.e. points where a seamless transition of control between the workflow engine and GDSS can be accomplished. Two different types are defined: (1) a *start* point (SP), which defines where a process starts; and (2) an *end* point (EP), defining where a process finishes.

Concerning a workflow process, we define that SP and EP are valid if the two transfer points completely enclose a set of tasks, as illustrated at the top of Figure 3. The bottom of Figure 3 illustrates invalid SP and EP. As it will be seen later, this restriction is necessary to identify a set of participants in a formal process by traversing forward or backwards the workflow procedure from SP to EP.

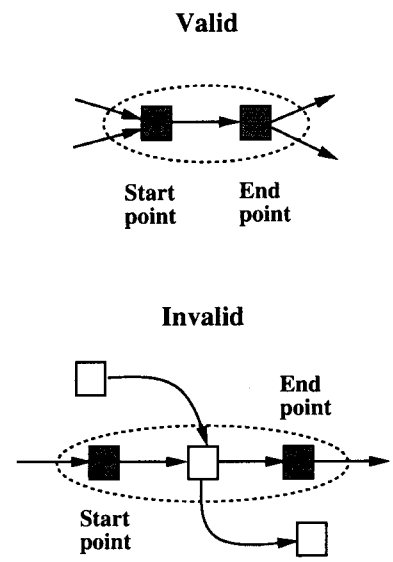

Figure 3: Valid/invalid start and end points for workflow procedures

A workflow EP may contain a condition. Exceptions to workflow procedures must use the EP conditions in order to preserve the enclose property described above. A large workflow process can be sub-divided in several parts, using

multiple transfer points. Also, workflow parts can be defined inside other parts.

Concerning a GDSS process, SP and EP define a time frame for participants in a group to interact informally. Within a time frame, phases for generating, discussing and voting ideas (at least one) may be multiply defined. The only restriction to SP and EP is that each phase within the time frame must not be broken. This restriction is only intended to simplify the implementation of an interface for the GDSS.

A GDSS EP may also contain a condition. We have found these conditions useful for handling improperly terminated informal processes (e.g. NGTOOL is a synchronous tool subject to network partitions).

3.2 Transfer Contexts

We assume that a workflow system circulates documents between users, which modify them privately (e.g. signing an equipment purchase) in order to accomplish required tasks. We say that the workflow system handles *tied* information, because it does not manipulate document contents. The documents received at the SP and delivered at the EP of a workflow procedure are designated *tied documents* (TD).

In the case of the GDSS, documents are not circulated between users, but users rather share them. We say that the system handles *untied* information, because it manages the document contents (fragmentation, replication, concurrency control protocols, etc.). The data objects shared by users at the start and end points of an informal process define, in our perspective, *untied documents* (UD).

Both TD and UD define which data enters and leaves processes, providing context to the transfer mechanism.

3.3 Transfer Actors

The set of individuals participating in tasks enclosed by a workflow SP and EP is designated *formal group* (FG). Similarly, the set of participants that interact in an informal process bounded by SP and EP is designated *informal group* (IG). Both FG and IG identify which actors are involved in a process transfer.

3.4 Transfer Mechanism

To transfer control from a formal process to an informal process one must interface the workflow engine EP with the GDSS SP. This interface does not only comprehend transfer points but also actors and contexts. So, the transfer mechanism must execute:

1. Untie the output document:
   $UD = Untie(TD)$
2. Informalize the group:
   $IG = Inform(FG)$
3. Enact the GDSS from the workflow end point:
   $SP = \mathcal{P}_i(EP, IG, UD)$

Inversely, to interface an informal process with a formal process, the transfer mechanism must:

1. Tie the output document:
   $TD = Tie(UD)$
2. Formalize the group:
   $FG = Form(IG)$
3. Enact the workflow engine at the GDSS end point:
   $SP = \mathcal{P}_f(EP, FG, TD)$

The mechanism described above resolves all but one of the limitations we found in the ORCHESTRA's mechanism: it merges workflow exception handling with planned process transfers, using the conditional end points; and it identifies the actors involved in each process.

4 Transforming Formal and Informal Processes

The transformation of a formal process into an informal process presents several opportunities for organizational development. One is that it empowers people with more decision-making abilities, which results in groups more readily prepared to changes and uncertainties. Another is that it decentralizes power and knowledge, and thus avoids risks associated to key personnel. Also, the satisfaction with group work should not be neglected. Nevertheless, informal processes present several drawbacks: coordination effort,

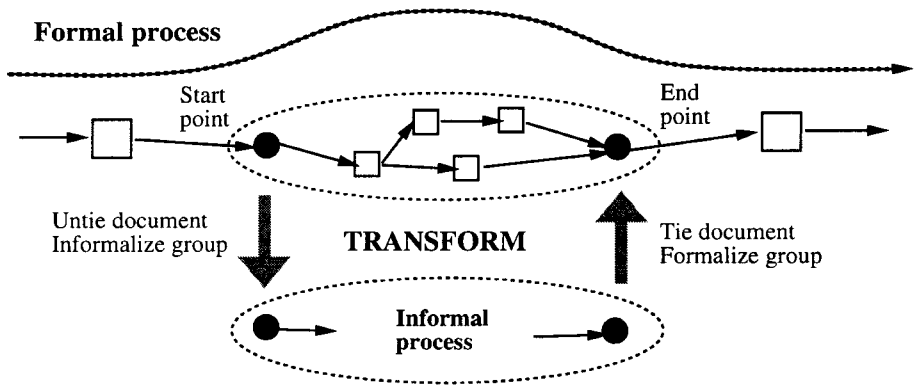

Figure 4: Informalize part of a formal process

extra time spent in decision-making or handling personality problems, groupthink, choice shifts. Therefore, the best approach is to only apply transforms to small parts of formal processes.

Our strategy is the following. Two transfer points must be defined for transforming a formal process. The transfer context is preserved for the start point, i.e. the document which would be delivered to a participant in a workflow procedure is instead delivered to the GDSS, and also for the end point, delivering the GDSS results to the participant identified by the end point. The actors involved in the formal process, which would execute individual tasks coordinated by the system, will be involved in the informal process and requested to cooperate in a group task. The group decides how and when to finish the informal process, knowing the results expected at the formal process's end point.

Observe Figure 4, the transformation of a formal into an informal process requires:

1. Untie the output document:
   $UD = Untie(TD)$
2. Informalize the group:
   $IG = Inform(FG)$
3. Enact the GDSS:
   $UD = \mathcal{P}_i(EP, IG, UD)$
4. Tie again the results from the GDSS:
   $TD = Tie(UD)$
5. Formalize back the group:
   $FG = Form(IG)$
6. Proceed, enacting the workflow engine:
   $TD = \mathcal{P}_f(EP, FG, TD)$

The transformation of an informal process into a formal one should not only be possible but desirable. A formal process releases coordination effort from people, turns individual responsibility and commitment straightforward and allows tight control over task schedule and accomplishment. Most importantly, this type of transformation gives visibility to tasks which are not explicit in the organization. Therefore, the following transformation is also defined:

$$\textit{Tie}\ (\text{UD}) + \textit{Form}\ (\text{IG}) + \mathcal{P}_f(EP, FG, TD) + \textit{Untie}\ (\text{TD}) + \textit{Inform}\ (\text{FG}) + \mathcal{P}_i(EP, IG, UD)$$

5 Implementation

SP and EP The first implementation step concerns the definition of interfaces between the transfer mechanism, workflow engine and GDSS. We decided to adhere to the interface defined by the Workflow Process Definition Language (WPDL). WPDL derives from an effort of the Workflow Management Coalition (WfMC) [10], a non-profit international body that, since August 1993, attempts to standardize workflow interoperability and connectivity between workflow products. WPDL characterizes the type of data that can be exchanged with a workflow engine. Note, in the following example, the activity and transition lists, and participants identification:

```
WORKFLOW          ''At the Sales Department''
  WPDL_VERSION    1.0
  VENDOR          Vendor:Product:Release
  CREATED         1995-12-06
// <Activity List>
// <Transition List>
END_WORKFLOW

PARTICIPANT       ''Tim White''
  TYPE            HUMAN
  USERID          ''tw456''
  SURNAME         ''White''
```

```
  FORENAME          ''Tim''
  DESCRIPTION       ''Mail Clerk''
END_PARTICIPANT

PARTICIPANT         ''Presidents_secretary''
  TYPE              ROLE
  PERSONDESCR       ''France Baroque''
  DESCRIPTION       ''handles presidents mail''
PARTICIPANT

APPLICATION         scan_document
  TOOLNAME          winscan.exe
  OUT_PARAMETERS    scanned_document
END_APPLICATION

DATA                document_type
  TYPE              string
  DEFAULT_VALUE     ''Sales Order''
END_DATA
```

We use exactly the same WPDL interface for the GDSS, although the transition list has a different semantics: it is used to identify decision phases (typically, brainstorming → discussion → voting).

Workflow SP and EP are identified by the first and last elements of the transition list in a WPDL interface.

Process Enactment Before enacting a process, the transfer mechanism verifies if the SP, which is also a EP, has a condition. If a condition exists, it is processed to determine which alternative process is enacted.

Afterwards, a formal process is enacted by generating a transition event with FD/SP information in the workflow engine. An informal process is enacted differently. One instance of the GDSS is started at each IG's site with IG/SP information loaded[5]. Furthermore, the ID is replicated to each participant and a message of type "please discuss document ID" is loaded to the GDSS. Currently, we do not select which decision phases are executed by the GDSS.

Transfer Contexts The generation of a TD, to be loaded to the workflow engine, is implemented by the *Tie(UD)* function. Currently, this function only fills in the data type field of the WPDL interface. The generation of a ID, implemented by the *Untie(TD)* function, corresponds to the replication of the TD to the IG, as explained before.

Transfer Actors The identification of a IG in a workflow procedure is implemented by the *Inform(FG)* function. Given that SP and EP are defined by a WPDL interface, this function extracts the list of participants from the WPDL. It recursively goes through the activity list, searching for other workflow procedures and associated participants. The enclose property, imposed to valid SP and EP, ensures that a finite set of participants is identified.

The identification of a FG, assigned to the *Form(IG)* function, is unnecessary, given that, to enact the workflow engine, only the SP must be identified. This function is void.

[5] We use a replicated approach, connecting each replica with TCP/IP sockets. See [1] for details.

6 Conclusion and Future Work

Our approach addresses the integration of formal and informal work processes in organizations. The required interfaces and transition points are defined to allow seamless control patterns traversing different process types.

Transformations are proposed to switch work modes, formalizing or informalizing processes. Formal processes, subject to multiple interruptions due to erroneous definitions of procedures or organizational changes, may become informal, which guarantees that tasks may be accomplished based on mutual adjustments between individuals. Informal processes can become formal, releasing users from coordination efforts and establishing commitments.

The proposed transformations change two work characteristics: the nature of the group of individuals and the nature of information associated to processes. Formal processes are characterized by formal groups and tied data, while informal processes are characterized by informal groups and untied data. Therefore, a transformation is accomplished by simultaneously transforming the nature of group and data.

The implementation of the proposed mechanism allows multiple options for supporting informal processes: synchronous, asynchronous, highly structured, loosely structured, mediated or not.

The current status of our approach still presents some limitations: the requirements over workflow start and end points limit the cases where transformations can be applied; the decomposition of transformations was not addressed; and we have not found an automatic solution for transferring semantic contents between processes.

Furthermore, we have not studied how transitions or transformations can be requested on-the-fly by participants: one participant in a formal process would request the system to transform that process into an informal one, or the participants in an informal process would request that the process be substituted by a formal one.

Acknowledgments

The ESPRIT Project ORCHESTRA was funded by the European Community. The authors wish to acknowledge the other partners in the project: INESC, SMD, Novabase, Software AG, Taskon AS, EDP, Edinfor, CN, UITESA, CONUC, Lisle, ISCTE and UPM.

References

[1] P. Antunes and N. Guimaraes. Structuring elements for group interaction. In *Second Conference on Concurrent Engineering, Research and Applications (CE95)*, Washington, DC, August 1995. Concurrent Technologies Corporation.

[2] P. Antunes and N. Guimaraes. User-interface support for group interaction. In *Second CYTED-RITOS International Workshop on Groupware CRIWG '96*, Puerto Varas, Chile, September 1996. CYTED-RITOS.

[3] P. Antunes, N. Guimaraes, J. Cardenosa, and J. Segovia. Beyond formal processes: Augmenting workflow with group interaction techniques. In *ACM Conference on Organizational Computer Systems COOCS '95*, San Jose, California, August 1995. ACM.

[4] J. Davis and V. Hinsz. Current research problems in group performance and group dynamics. In H. Brandstatter, J. Davis, and G. Stocker-Kreichgauer, editors, *Group Decision Making*. Academic Press, Inc., 1982.

[5] R. Johansen, D. Sibbet, S. Benson, A. Martin, R. Mittman, and P. Saffo. *Leading Business Teams*. Addison-Wesley, 1991.

[6] H. Mintzberg. *The Structuring of Organizations*. Prentice-Hall, 1979.

[7] H. Mintzberg. A typology of organizational structure. In Ronald M. Baecker, editor, *Readings in Groupware and Computer-Supported Cooperative Work*, pages 177–186. Morgan Kaufmann Publishers Inc, 1993.

[8] Orchestra. The orchestra project, 1996. N. Guimaraes (ed). Springer-Verlag.

[9] T. Reenskaug, P. Wold, and A. Lehne. *Working with Objects: The Ooram Approach Software Engineering Method*. Manning Publications, 1996.

[10] WfMC. Workflow management coalition interface 1-5, 1996.