Internet Based Inter-Business Process Management: A Federated Approach

Giacomo Piccinelli, Floriana Marcello*, Gabriele Zugliani*
Internet Business Management Department
HP Laboratories Bristol
HPL-98-87
April, 1998

E-mail: giapicc@hplb.hpl.hp.com
marcello@cli.di.unipi.it

workflow,
internet-process,
federation

EDI (Electronic Data Interchange) was a powerful business to business (B2B) process enabler well before the explosion of the Internet. The competitive advantage deriving from this practice is quite clear as EDI had a big impact in the streamlining of administrative processes and business EDP (Electronic Data Processing) in general. The main problem was the cost, deriving especially from networking infrastructure, which prevented small and middle-size businesses accessing this resource.

Thanks to the low-cost connectivity and the popularity reached by the Web in terms of electronic commerce (EC) enabler, a lot of businesses have now a basic infrastructure needed to support B2B processes and this is the point from which our work moves.

As part of an ongoing project on federated processes, we present the results obtained with RABBIT (Research on Advanced B2B Information Technology): an infrastructure for the definition and enactment of federated – distributed processes. The architecture allows the specification of multiple-organisation to process independently from the geographic distribution of the organisations involved while the coupling of a specific compilation technology and execution engines supports the distributed enactment of the process. Basic network-fault tolerance and privacy are supported at process level.

* Computer Science Department, University of Pisa, Pisa, Italy

Internet Based Inter-Business Process Management: A Federated Approach

Giacomo Piccinelli

Extended Enterprise Laboratory
Internet Business Management Department
Hewlett-Packard Laboratories
Bristol BS12 6QZ, U.K.

Email: giapicc@hplb.hpl.hp.com

Floriana Marcello and Gabriele Zugliani

Computer Science Department
University of Pisa
56100 – Pisa
Italy

Email: marcello@cli.di.unipi.it

Abstract

EDI (Electronic Data Interchange) was a powerful business to business (B2B) process enabler well before the explosion of the Internet. The competitive advantage deriving from this practice is quite clear as EDI had a big impact in the streamlining of administrative processes and business EDP (Electronic Data Processing) in general. The main problem was the cost, deriving especially from networking infrastructure, which prevented small and middle-size businesses accessing this resource.

Thanks to the low-cost connectivity and the popularity reached by the Web in terms of electronic commerce (EC) enabler, a lot of businesses have now a basic infrastructure needed to support B2B processes and this is the point from which our work moves.

As part of an ongoing project on federated processes, we present the results obtained with RABBIT (Research on Advanced B2B Information Technology): an infrastructure for the definition and enactment of federated - distributed processes. The architecture allows the specification of multiple-organisation to process independently from the geographic distribution of the organisations involved while the coupling of a specific compilation technology and execution engines supports the distributed enactment of the process. Basic network-fault tolerance and privacy are supported at process level.

1 Introduction

It is difficult improving what we can not measure but it is extremely difficult measuring what we can not handle and process-based infrastructures offer a natural support for managing problem complexity. The evolution of new technologies [15,16] dramatically impacts on the way in which processes are designed and a lot of emphasis is on the integration of different components into global environments [16] and then into global processes [6,10,18]. Internet technology offers an unprecedented interconnection capability [11] that distributed object architectures [13,15,16] exploit in order to boost the creation of domain-focused and location-transparent environments [4,17]: the cooperation and coordination aspects become crucial [1,2,3].

In this scenario, a new layer of management (PCE[2]) is needed in order to support projects involving multiple competencies and the added value is in the process

---

[2] We use the term PCE to indicate both the environment and the system that manages the definition and enactment of a process in that environment.

coordination. We refer to this scenario as *federation* [3,5] and in this context we locate our work. Focusing on the coordination and information exchange aspects, we present a system for the definition and enactment of federated processes involving different organisations and/or different parts of the same organisation. After a brief overview of the more popular distributed object architectures and their impact on PCEs, we present and discuss the cooperation paradigm we enforce. The process-definition formalism is presented together with the federation basic infrastructure, PCE interfacing solutions and deployment facilities.

2 Distributed Object Architectures

Object models like DCOM (Distributed Component Object Model) by Microsoft [16], the OMA (Object Management Architecture) by OMG (Object Management Group) [13] and Java RMI (Remote Methods Invocation) [15] enforce two major aspects of an application: strong modularization (components) and location transparency.
Although location transparency is quite important for application components, the big impact of distributed object technologies on process-centered environments depends also on the "automation" [16] features they introduce. The mechanisms may be slightly different but the result is the same: applications may ask other applications to perform task and/or to supply data. Extra layers are built on top of basic architectures (like OLE - object linking and embedding – for DCOM or Common Facilities in the OMA [14]) in order for the applications to offer service-oriented interfaces.
In terms of the actual infrastructure we build to support the federation process, we focus on Java and RMI basic services.

3 Cooperation Model

The purpose of a cooperation process is to organise resources from different organisations[3] in order to achieve a common goal. The peculiar aspect of a *federation* [3,5] is the fact that a pool of independent and autonomous organisations agrees on a common process and the members share part of their resources and expertise in order to enact such a process. Despite the commitment to the common goal and the need to exchange data and services, *autonomy* and *secrecy* are fundamental issues for the members of a federation and any infrastructure that targets federative process support has to deal with these requirements.

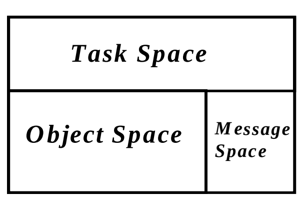


Fig.1: Workspace component

[3]We refer to a generic interpretation of the term *organization* indicating an autonomous and independent entity [5,9]. The term also indicates the set of applications associated with this entity.

We propose a solution based on the paradigm of a common workspace. Every organisation is associated with a part of this space called *workspace component* (Fig.1) representing its interface to the federation and the union of the workspace components represents the *federation workspace* (Fig.2). In its *object space*, an organisation puts the data it needs to share with its partners and it can retrieve data produced by its partners and relevant for the execution of its tasks. An organisation has immediate access only to the data in the object space of its W and these are the only data exposed to the federated process: autonomy of the organisations is preserved. Each organisation shares all and only the data it agreed to release and under the circumstances defined in the federated process. At the same time each organisation receives all and only the data it is entitled (requested) to work on. Objects are the result of an activity ("artifact" or "work item" [9]), messages represent information on the state of either the system or the process. Tasks are atomic operations like the execution of an activity or the manipulation (insert, withdraw, process) of data and messages.

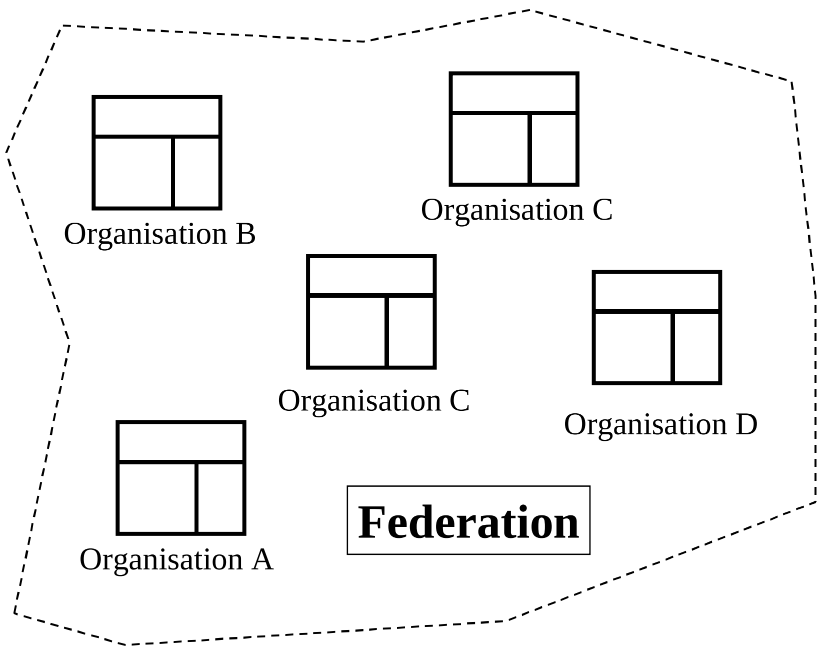


Fig .2: Federation workspace

The purpose of a federation infrastructure (F) is to manage the federation workspace in a way that, at any time, each organisation knows exactly what to do and has available the resources it needs. As F cannot interfere with the internal PCE of any organisation: autonomy is preserved.

3 Process Definition Language

The basic operations in a cooperative process are related to the exchange of artifacts, the exchange of synchronisation (control) information and the execution of activities related to internal tasks or supporting the work of other members of the federation (Tab.1). The value added by a process-based organisation depends on the fact that atomic components may be organised into complex activities (Tab. 2,3).

Push (OrgA**,** OrgB**,** Obj**)** **Pull (**OrgA**,** OrgB**,** Obj**)** **Message (**OrgA**,** OrgB**,** Msg**)** **Service (**OrgA**,** OrgB**,** Srv**,** Obj**)** **Task (**OrgX**,** Act **)**

Tab.1: Basic Operations

The influence on the formalism coming from languages like Hoare's CSP [8] and Milner's CCS [12] is quite strong but we explicitly target the peculiarities of a federated process instead of working with generic distributed processes. The actual semantic of the entire language has been formalised following an approach (operational style) similar to the C-FAM (concurrent functional abstract machine) used for FACILE [7]. The point of view taken during the design of a process is the one of an impartial coordinator that looks at the members of the federation as resources to organise in order to achieve a specific result. An organisation may supply (push) data, send them control information (messages) and asking (pull) for data. A task is related to an aspect of the process it is immediately responsible for but, in order to support the central role of cooperation in the federation, it may also be asked to help one of its partners (service).
The *sequential* operator ";" indicates that all the tasks in the process P_1 need to be completed before starting any task indicated in P_2: the overall process ends when P_2 ends.

$\mathbf{P_1 ; P_2}$	Sequential Composition
$< P_1$ **&** … **&** $P_n >$	Parallel Composition
(expr**)** **[**P_1 + … + P_n**]**	Choice Operator
Nil	Null process

Tab. 2: Composition Operators

The *parallel composition* operator allows multiple execution threads within a process while *choice* operator executes one and only one process among the P_i depending on the state of the federation. Procedures (Tab.3) are introduced mainly for modularization purposes but they also offer the possibility to specify recursive process definitions (ex. loops). The types we allow for the variables are: *org* (organisation), *msg* (message), *obj* (object), *act* (activity/task) and *srv* (service). We enforce a "late" evaluation policy concerning procedure-call evaluation and it is therefore possible to have simple as well as mutual recursion in the definitions.

Label**(**Var1:T1**,..,**VarN:Tn**)** **{** P **}**	Procedure definition
Label**(**Val1**,..,**ValN**)**	Procedure call

Tab. 3: Procedures

4 Federation Infrastructure

Main components of the support infrastructure for the implementation of a federated process are the compiler, the enactment engine(s) and the interface wrappers.

4.1 Compiler

For each organisation we build the process V_{org} that contains the specification of all and only the tasks the organisation is requested to do and the synchronisation points it has to maintain with respect to its partners. Basic operations are easy to map into V_{org} while the synchronisation problems come with the composition operators. The problem we have, for example with sequential composition, is pictured in the following example:

< A(xx) & B(xx) > ; < A(yy) & B(yy) >

If *xx* is completed in A but B is still working on it, A has to wait until also B completes *xx* before to start working on *yy* in order to preserve the semantics of the language. The compiler manages these situations with specific solutions that assure the intended semantic of the global process is preserved. This organisation-centric approach allows a modular organisation of the enactment infrastructure with major benefits also in terms of autonomy and security as well as fault tolerance. An organisation may follow its own process, unless explicit synchronisation points are specified, independently from other members of the federation (autonomy). Security is enforced by the fact that the compiler is consistent with the PSL semantics and all the actions an organisation is requested to perform derive from common federated process definition. Benefits from a fault tolerance perspective derive from the autonomy of the organisations: if an organisation experiences (temporary) problems its partner may not be affected.
We anticipate that the result of the compilation is location independent, which means we model independent components but information on the physic location of the organizations (components) is ignored at this stage.

4.2 Enactment Engines

In the enactment infrastructure we distinguish three main components (Fig. 4): workspace components (W), engines (E) and the interconnection support. Focusing on a single organization, the engine has complete access to its workspace component and it can also communicate with other engines but, in a normal situation, it cannot interact directly with any PCE. Each engine Ex enacts the projection Px of the federated process produced by the compiler for the organization X and its main job is related to messages and data management, task posting and synchronization. Also for the engine implementation, the complexity is concentrated in the support for multiple execution threads, sequential integrity and choice-step consistency.

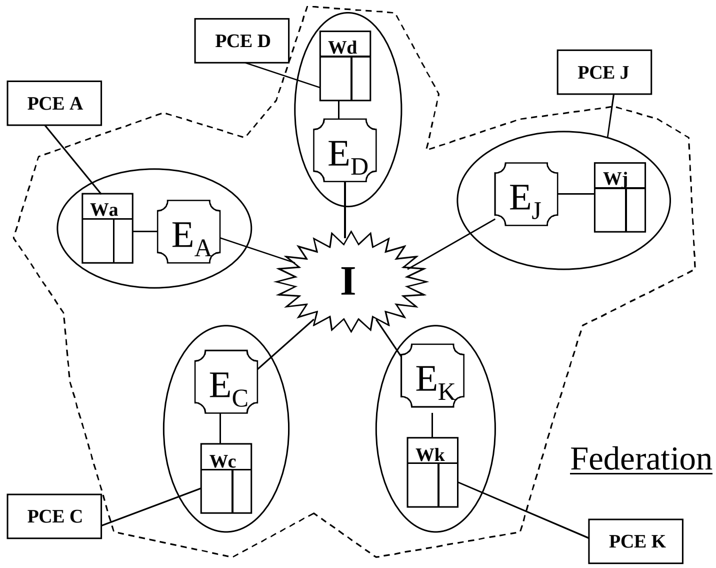


Fig.4: Enactment Infrastructure

Choice-step consistency problems, for example, depends on the fact that if a path (Pi) is chosen, within a choice operator, for one of the projections of the global process then also in the enactment of all other projections we need to follow the same path. Major issue is that we allow different execution speed in different organisations and, in order not to introduce implicit synchronisation points (with solutions like waiting for all the organisation involved in the choice to reach the evaluation point), specific solutions need to be enforced both in the engine and in the compiler.

4.3 PCE Interface

The PCE of an organisation needs a bridge to the W in order: (1) to put and get messages and data as indicated by the tasks posted by the process and (2) to access the indications on the tasks it has to perform. The W is mainly a container of data and information, and the bridge to the PCE depends on the level of interactivity and automation it enforces. In our investigation we focused on two extremes (full automation and pure presentation) but solutions in between are also possible. We focused on were Java and CORBA technology though OLE is also under investigation.

5 Deployment

The main components of our architecture (enactment aspect) are the engines, the workspace components and wrappers and thanks to the support of Java RMI (remote methods invocation) we enforced the possibility to allocate all of them in different ways without major changes.

6 Conclusions

Distributed object architectures (DCOM, CORBA, Java RMI) coupled with Internet and Intranet technology have a great impact in process-centered environments both in terms of connectivity and application automation.

We present a complete infrastructure supporting the federated process starting from its definition to its actual enactment. Few simple basic operators and the possibility to build high-level modules and process libraries represent the design environment we provide while the enactment environment is based on the result of a distribution-oriented compiler and specific cooperation environment. Concerning the deployment of the federation infrastructure, the components are built taking into consideration location transparency problems therefore we can tune the deployment process on the peculiarities of the federation. Autonomy, security and fault tolerance issues are reflected in all the choices and actual components in our architecture.

Bibliography

[1] S. Bandinelli, E. Di Nitto and A. Fuggetta. Supporting cooperation in the SPADE-1 environment. In *IEEE Transactions on Software Engineering,* Vol. 22, no. 12, December 1996.

[2] N.S. Barghouti. Supporting cooperation in the Marvel process-centered SDE. In *Fifth ACM SIGSOFT Symposium on Software Development Environments.* Herbert Weber (ed.), 1992.

[3] C. Basile, S. Calanna, E. Di Nitto, A. Fuggetta and M.Gemo. Mechanisms and policies for federated PSEEs: basic concepts and open issues. In *Proc. 5th Europeen Workshop on Software Process Technology*. Nancy, France, 1996.

[4] I.Z. Ben-Shaul, A. Cohen, O. Holder and B. Lavva. HADAS: A Network-centric framework for interoperability programming. In *Proc 2nd Inter. Conference on Cooperative Information Systems.* June 1997.

[5] I.Z. Ben-Shaul and G.E. Kaiser. Federating process-centered environments: the Oz experience. In *Automated Software Engineering*, Vol. 5. Kluwer Academic Publisher, 1998.

[6] I.Z. Ben-Shaul and G.E. Kaiser. Integrating groupware activities into workflow management. In *Proc. 7 th Israeli Conference on Computer Based Systems and Software Engineering.* June 1996.

[7] Giacalone, P. Mishra and S. Prasad. FACILE: A symmetric integration of concurrent and functional programming. In *Proc. of TAPSOFT'89,* Vol.2. Lecture Notes in Computer Science (LNCS 352). Springer-Verlag, 1989.

[8] C.A.R. Hoare. Communicating Sequential Processes. *Series in Computer Science*. Prentice-Hall, 1985.

[9] D. Hollingsworth. The workflow reference model. Workflow Management Coalition (WfMC), TC00-1003, November 1994.

[10] N. Krishnakumar and A. Sheth. Managing heterogeneous multi-system task to support enterprise-wide operations. In *Distributed and Parallel Databases.* Kuwler Academic Publishers, 1995.

[11] J. Miller, A. Sheth, K. Kochout and D. Palaniswami. The future of Web-based workflow. In *Proc. of the International Workshop on Research Directions in Process Technology*. Nancy, France, July 1997.

[12] R. Milner. A calculus of communicating systems. *Lecture Notes in computer Science* Vol. 32. Springer-Verlag, 1980.

[13] Object Management Group (OMG). A discussion of the object management architecture. January 1997.

[14] Object Management Group (OMG). CORBA facilities: common facilities architecture V4.0. November 1995.

[15] R. Orfali and D. Harkey. Client/Server programing with Java and CORBA. Wiley Computer Publishing, 1997.

[16] Creating Programmable Applications with OLE Automation. Vol. 1 and 2. Microsoft Press, 1994.

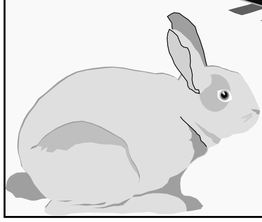
[17] A. Sheth (ed.). Report from the NSF workshop on workflow and process automation in information systems. In *Proc. NSF workshop on workflow and process automation in information systems: state-of-the-art and future directions.* A. Sheth (ed.), May 1996.

[18] G. Valetto and G.E. Kaiser. Enveloping sophisticated tools into process-centered environments. In *Proc. 7th IEEE International Workshop on CASE.* IEEE Computer Society Press, 1995.

Appendix: Poster Presented at HP OVUA'98 (Rennes)

Research on Advanced
Business to Business
Information Technology

RABBIT



G. Piccinelli

F. Marcello and G. Zugliani

Project Objectives

EDI (Electronic Data Interchange) has been a powerful business to business (B^2B) process enabler well before the explosion of the Internet. The competitive advantage deriving from this practice is quite clear as EDI had a big impact in the streamlining of administrative processes and business EDP (Electronic Data Processing) in general. The main problem was the cost, deriving especially from networking infrastructure, which prevented small and meddle-size business to access this resource.

Thanks to the low-cost connectivity and the popularity reached by the Web in terms of electronic commerce (EC) enabler, a lot of businesses have now the basic infrastructure needed to support B^2B processes and this is the point from which our work moves.

As part of an ongoing project on federated processes, we present the results obtained with RABBIT (Research on Advanced B^2B Information Technology): an infrastructure for the definition and enactment of federated - distributed processes.

The architecture allows the specification of multiple-organisation process independently from the geographic distribution of the organisations involved. The coupling of a specific compilation technology and execution engines supports the distributed enactment of the process.

Internet Business Chain

Distributed Object Technology

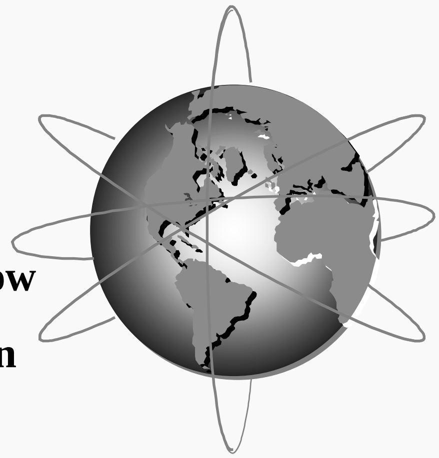
CORBA

RMI
(Remote Method Invocation)

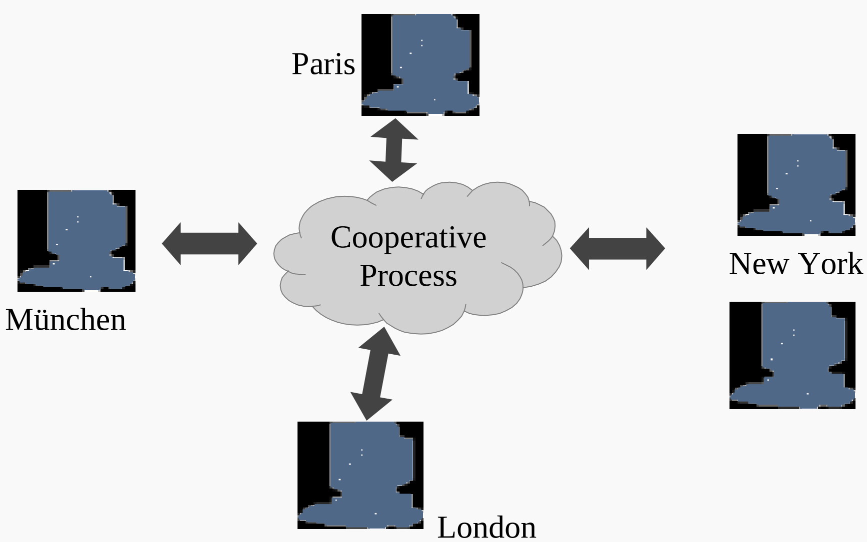
DCOM

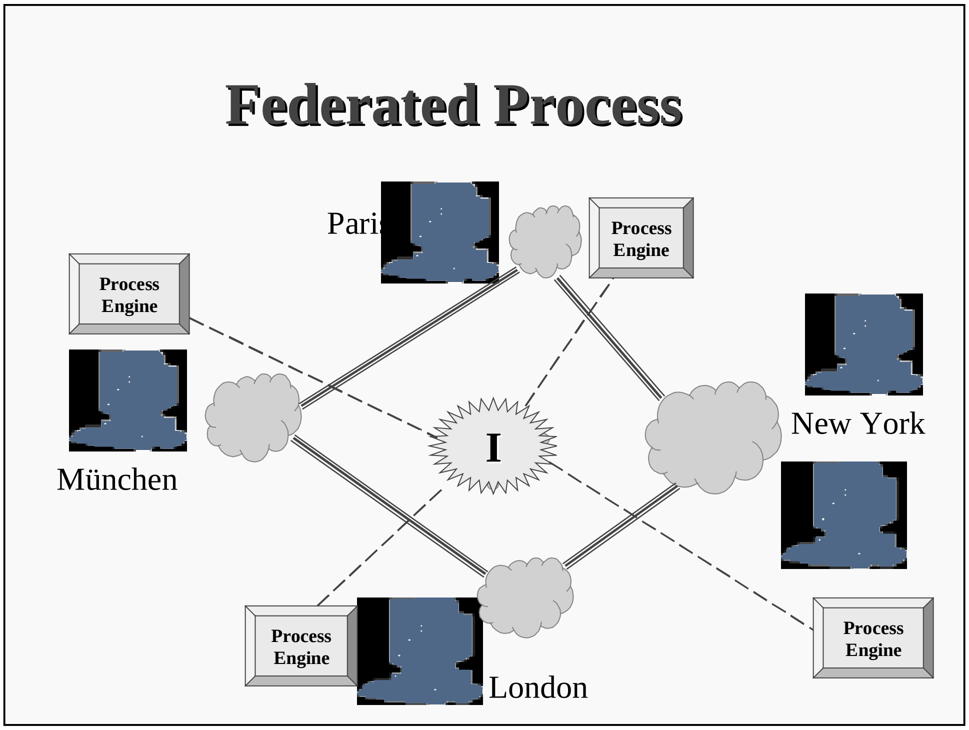
Internet Process Support

- **Connectivity:**
  - **External**
  - **Internal**
- **Electronic Data Flow**
- **Process Automation**



Cooperative Process




Federated Process
Paris
Process Engine
Process Engine
München
I
New York
Process Engine
Process Engine
London