Business process Eclipse Editor (BEE)

Paolo Maresca, Armando Cotugno, Salvatore Mignogna,
Roberto Longobardi*, Alessandro Donatelli*, Rosario Gangemi*

DIS - Dipartimento di Informatica e Sistemistica – University of Naples "Federico II"
* IBM Software Group Lab, Roma
paomares@unina.it, {armacot, salvatoremignogna}@gmail.com,
{roberto.longobardi, alex.donatelli, rosario.gangemi}@it.ibm.com

Abstract. *An important aspect of business process modelling is let non-professional users understand business processes, so that they can be involved not only in their understanding but even in their management. This paper introduces a business process editor geared to end users whose prerequisites are very elementary. The editor has been developed by the DIS of University of Naples "Federico II", in a partnership with the IBM Software Group Lab in Rome. The application has been developed on Eclipse Ganymede platform, and the first release is currently being tested at DIS. Cooperative design and development was made using IBM Rational Team Concert [1].*

Keywords. Business Process Modelling, Process Driven Development, USBD, Eclipse, Graphical Modeling Framework, IBM Rational Team Concert

Introduction

For long time only professional and highly proficient users have been able to effectively contribute to business process engineering. For such reason business process development tools became more and more difficult to use. Building an editor tailored on business process end users experience provides a good chance to get them involved in process management too. Business process Eclipse Editor (BEE) is a feature for Eclipse [2] developed using the Graphical Modeling Framework (GMF) [3]; BEE main goal is to provide ability to model business process for requirements elicitation. The idea behind BEE stems from a research done by IBM Software Group Lab in Rome which led to the establishment of a new method of software development: Unified Scenario-Based Design (USBD) [4]. This methodology is intended to bring the developed software to the real needs of business it will support; to achieve this goal, USBD proposes, among other things, a *process driven* [5] approach to software development, which states that software requirements should be derived from a formal definition of business process that the software must support. In this context, it was launched a collaboration between IBM Software Group Lab in Rome and the University of Naples "Federico II" with the latter engaged in a work of research and development within the business process modelling field: the current

version of BEE is the latest achievement in this work. Design and development of Business process Eclipse Editor was preceeded by two researches. The first one, *Comparing Business Definition Languages* [6], thoroughly explores the universe of languages and notations used to describe business processes, emphasizing the vastness and highlighting how the more widespread notation, and at the same time the most semantically complete one, is Business Process Modeling Notation (BPMN) [7]. The second research, *Comparative assessment of open source applications for business process management* [8], analyzes some applications in Eclipse environment for business process modelling, concluding that there are two types of applications: those that model workflow, therefore targeted at IT-professionals, and those describing business and its processes in a detailed way in order to support analysis and reengineering of processes themselves, therefore targeted at strictly business consumers. No business process modelling tools, especially open source, try to fill the business-IT gap providing requirements elicitation facilities. In this scenario we sensed the usefulness of developing an open source tool that, as suggested by USBD methodology, used business process modelling to obtain requirements of software that supports business processes themselves. In order to allow using tools to users that are novice in modelling business and in information technology either, it has been developed an application that provides the ability to fully model business processes, without being too complex to use. We needed to refer a business process ontology specifically designed for the Business process Eclipse Editor, which retained only the basic concepts to describe a business process, had those needed to use the business process models for requirements elicitation, and was also simply extensible for future developments. Because BPMN is so well known, it has been used a notation inspired by it but simplifying most of its complex aspects. Relying upon result of researches mentioned hereinbefore, our main effort has been on defining an ontology, hence a metamodel having the previously mentioned features, and then later used as a starting point to design the BEE model driven development. The development of BEE was conducted using the collaborative development tool IBM Rational Team Concert; this application, exploiting the potential of the framework IBM Rational Jazz Platform [9], facilitated development process engineering, as well as ideas and artifacts sharing in a team of displaced workers. Last version of the Business process Eclipse Editor allows to graphically model business processes, and to shape their business context. BEE keeps evolving; for the foreseeable future we expect to deploy new key features: requirements elicitation and traceability between business models and requirements of software supporting business processes. These characteristics suggest BEE as an innovative open source tool in strategic field of business process modelling for requirements elicitation.

BEE data metamodel

Ideal starting point to develop the metamodel to use in BEE was the List-Korherr metamodel [10], shown in figure 1. Relying on previous researches, Austrian researchers developed a metamodel by collecting concepts from both academic and industry fields. As result they had a simple but still complete metamodel, capable of

representing a very wide range of concepts concerning business processes and their own context.

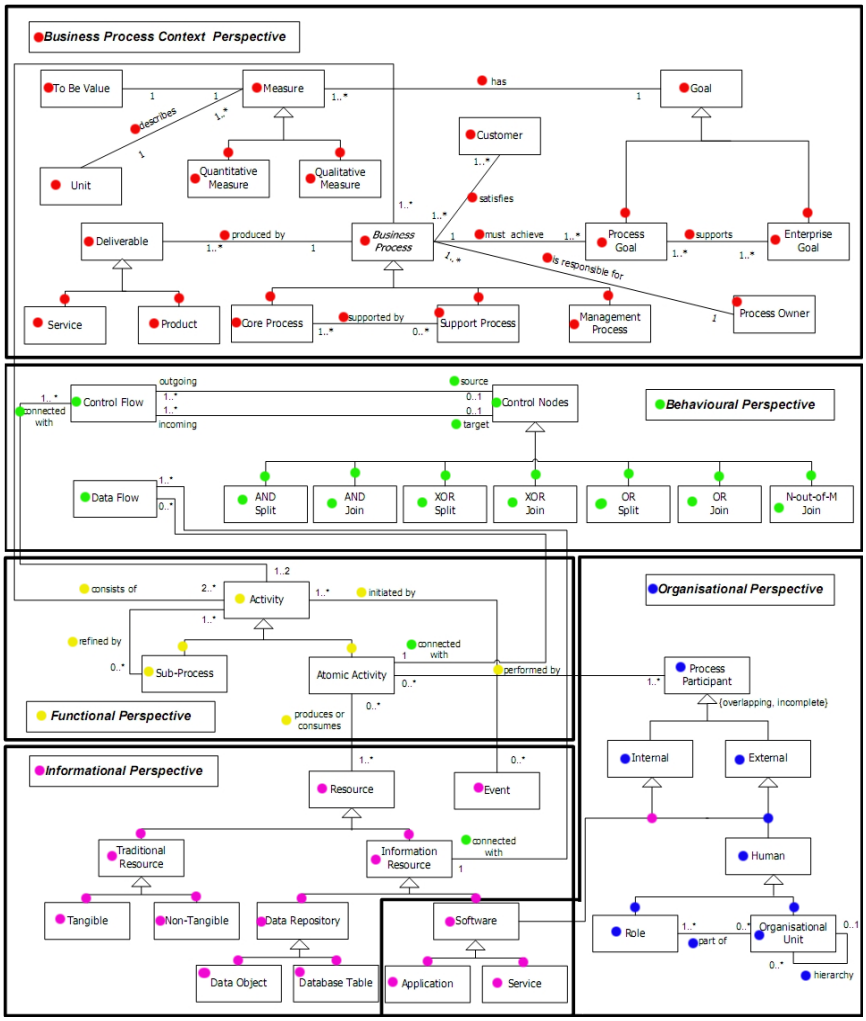

Fig. 1. List-Korherr metamodel

List-Korherr metamodel categorizes concepts in 5 *perspectives*: *functional perspective* brings together concepts related to the business process activities, while the *behavioural perspective* contains concepts about activity sequence through execution flows, about nodes controlling such flows, and about flow of information between activities; in the *informational perspective* we can find concepts pertaining the business process information content, depicted by the `Resource` hierarchy which describes the kind of resources that can be produced and consumed by process activities; *organizational perspective* encompasses concepts related to the business

organization in terms of organizational units and business roles; last one, *business process context perspective* brings together concepts relating to the business environment in which business processes live, such as business and process goals, goal measures, deliverables, customers information, process owner information.

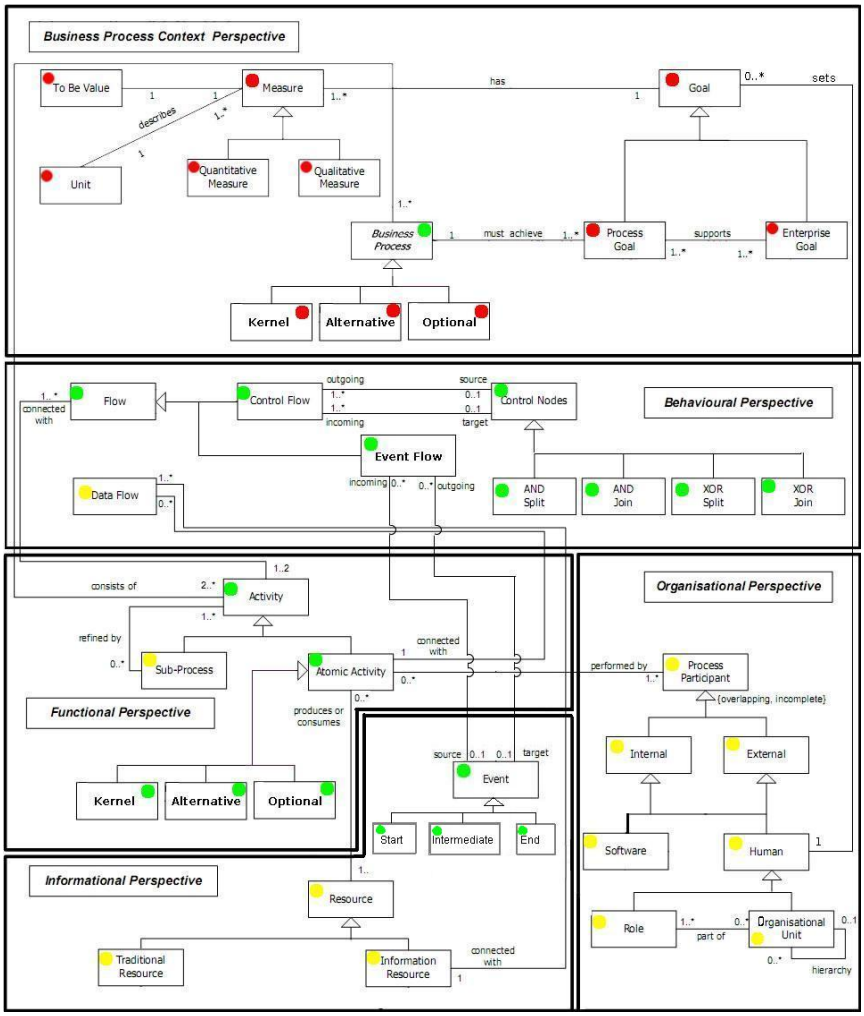

Fig. 2. Advanced List-Korherr metamodel

Starting from the List-Korherr metamodel, we proceeded at first by adding to it concepts that are typical of the USBD methodology and are needed to the main purpose of BEE (that is, business process modelling for requirements elicitation). We also eliminated concepts that was either unessential to the tool purposes or too semantically difficult to be accepted by an audience with no or little modelling skill; our work resulted in the Advanced List-Korherr Metamodel, shown in figure 2.

With regard to the *business process context perspective*, `Deliverable` and `Customer` concepts were removed as well as the `Core`, `Support` and `Management Process` specializations of `Business Process`: these concepts were considered of little interest in a tool whose main purpose isn't a description of every aspect of business processes and of the business environment surrounding them, but to obtain a business process definition that is formal, complete and simple at the same time. For the same reason, we simplified: the *informational perspective*, removing some specializations from the `Resource` hierarchy; the *organizational perspective*, simplifying the `Process Participant` hierarchy; the *behavioural perspective*, removing some of the control nodes (namely, `OR-Split/Join` and `N-out-of-M Split`) whose semantic is too complex. We want to point out that the latter change doesn't alter our metamodel expressive power, as the other control nodes together constitute a functionally complete set.

We also added the following concepts, that we considered important, to the List-Korherr metamodel:

- `Business Process`, as well as `Activity`, specialization into `Kernel`, `Optional` and `Alternative Process`, as specified by USBD methodology;
- `Event Flow`, and the `Flow` generalization, where `Flow` generalizes both `Control Flow` and `Event Flow`;
- `Event` specialization into `Start`, `Intermediate` and `End Event`, to augment the metamodel expressive power and to make usage of tool based on it simpler;
- relation between `Goal` and `Process Participant`, representing the concept "process participants set business and process goals".

The Advanced List-Korherr metamodel was in turn the starting point to obtain a new metamodel which could be concretely used in a model-driven development process, as the one supported by GMF which has been chosen for BEE development. The most significant change is a high-level reorganization of the metamodel; List-Korherr metamodel five perspectives was reduced to three:

- *business process*, that encompasses concepts originally present in *functional*, *behavioural* and (part of) *informational perspectives*;
- *business organization*, corresponding to the *organizational perspective*;
- *business context*, corresponding to the *business process context perspective*.

These perspectives, defined as metamodel classes, are aggregated into the `BusinessModel` class, that represents the whole business model.

Other important changes to the Advanced List-Korherr metamodel are the following:

- introduction of a `Node` class, that is a generalization of `Activity`, `Event` and `ControlNode` concepts: such generalization introduces a great simplification, and makes possible to consider only one flow, the `ControlFlow`, seen as a flow between *nodes*, so that we don't need to distinguish between different flow types connecting different elements;

- Sub-Process concept, originally included in the List-Korherr metamodel, is now represented considering `BusinessProcess` as a specialization of `Activity`; this choice is valid from a semantic point of view, since a similar definition is used both in BPMN 1.0 OMG specification (where `SubProcess` is a `Process` and `Process` is an `Activity`, hence `SubProcess` is an `Activity` too)[1] and in Eclipse STP BPMN metamodel[2], where `SubProcess` is an `Activity`; furthermore, our choice appears to be convenient from a practical point of view, since it will allow, once needed, to re-use the same graphical editor used for modelling a `BusinessProcess` to model the behaviour of a sub-process, using GMF diagram partitioning[3] facility.

We also added a boolean attribute, `ToBeAutomated`, to `Human` class: this is a key information for requirements elicitation, as explained hereinafter.

We decided to operate such changes for several reasons:

- BEE metamodel is easy to read: concepts partitioning into perspectives is clearly and formally defined; all perspectives are linked together in a unique `BusinessModel`;
- BEE metamodel brings to a well structured model editor: EMF generated model editor reflects metamodel structure, thus a well organized structure automatically results in a clear and easy to use tree representation of data in model editor;
- BEE metamodel well suits GMF editors generation: GMF graphical editor generation asks for a *root diagram element* to be specified; to keep diagram elements linked to the underlying EMF model, they need to be in a containment relation with such root element: defining a root element for each metamodel perspective thus facilitates independent generation of several GMF editors each based on a specific metamodel perspective;
- BEE metamodel is easy to extend: metamodel structure makes it easy to add new perspectives when necessary with little or no impact on other perspectives.

Currently, List-Korherr metamodel *informational perspective* isn't yet included in our metamodel and is expected to be deployed in the next BEE release. The resulting metamodel, which was used for BEE and constitutes its fundamental architectural component, is shown in figure 3.

---

1 BPMN 1.0 OMG Final Adopted Specification, http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf , pp. 16 and 19; a graphical representation of BPMN 1.0 metamodel has been developed by Wsper Group and is available at http://www.wsper.org/bpmn1.0.jpg

2 STP BPMN is a STP (SOA Tools Platform, http://www.eclipse.org/stp/) subproject; it's a business process graphical editor based on BPMN notation; STP BPMN metamodel graphical representation is available at http://www.eclipse.org/stp/bpmn/model/index.php

3 *Diagram partitioning* technique is explained in an Eclipse Wiki page available at http://wiki.eclipse.org/Diagram_Partitioning

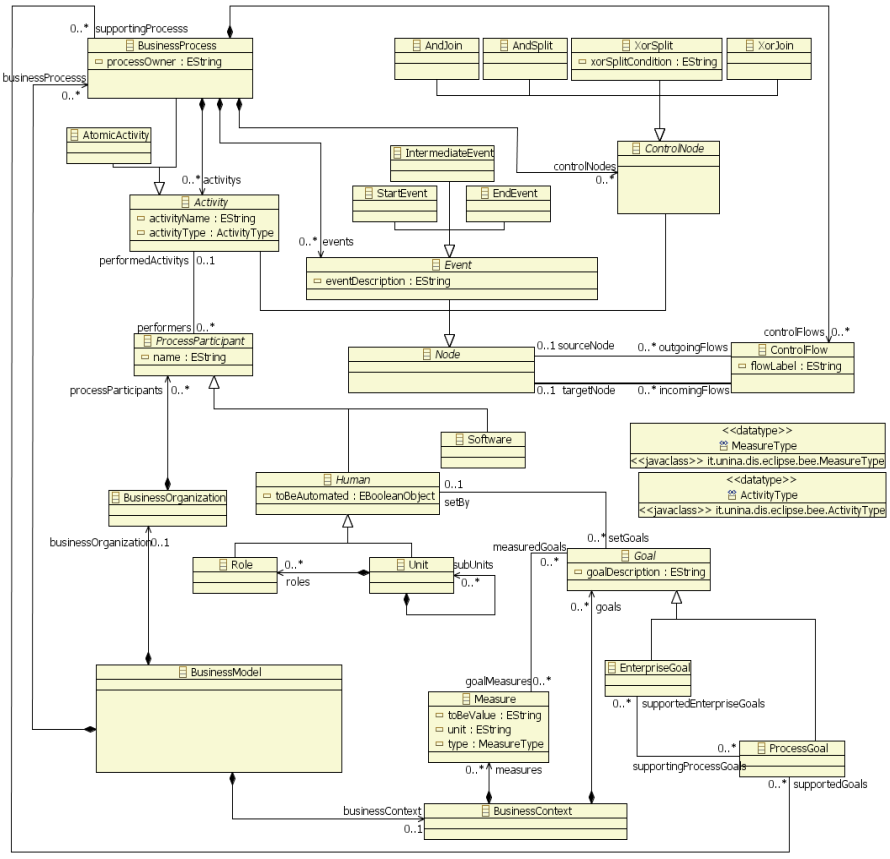

Fig. 3. BEE metamodel

BEE architecture

Graphical Modeling Framework (that integrates Eclipse Modeling Framework [11] and Graphical Editing Framework [12] capabilities) allows generating more than one graphical editor based on the same model: each editor allows working on a specific portion of the metamodel. The model editor (generated through the EMF component of GMF), instead, allows working on the whole metamodel. All editors (EMF model editor and GMF graphical editors) can be afterwards integrated in a single multipage editor. Best choice for the BEE project plan, therefore, was to schedule the development of a first graphical editor that allows modelling fundamental process concepts (those pertaining the *business process* in our metamodel), together with an EMF generated model editor that allows working on the entire metamodel (*business model*).

BEE project plan schedules a subsequent development of more graphical editors and, finally, the integration of all editors; metamodel structure we developed lends itself to this kind of development and then appears to be the best architectural choice.

BEE metamodel was used to develop, by mean of GMF, the Java code realizing:

- data model structure;
- data model access features;
- non graphical model editor;
- graphical editor.

Our development result consists therefore of four Eclipse plug-ins: two of them supply data model access features, while the other two realize, respectively, the (non graphical) model editor which allows working on the whole *business model*, and the graphical editor which allows working, by mean of graphical tools, on model elements pertaining the *business process perspective*. The Business process Eclipse Editor architectural structure is shown in figure 4.

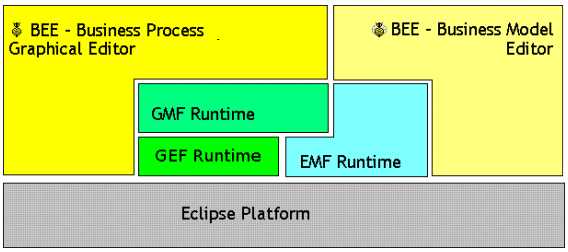

Fig. 4. BEE Architecture

BEE collaborative development through IBM Rational Jazz

IBM Rational Team Concert has been a key tool to develop the Business process Eclipse Editor (BEE). This tool is based on IBM Rational Jazz Platform. We have been testing and using client and server components of Jazz framework since their first beta release.

BEE development team was formed by three IBM Software Group Lab researchers and two students of the University of Naples “Federico II”, supervised by professor Paolo Maresca. Since such development team was so heterogeneous and spread across different locations it was necessary to employ a tool that allowed remote working and cooperation. With Jazz we have been coordinating project work, scheduling and planning all project development steps, sharing and defining rules and responsibilities of the project development. Jazz Work-Items mechanism allowed us to control and manage project progress, monitor team members' work, assign tasks in a clear way and define tasks time constraints. Jazz is a very complex platform that supports each phase of software development cycle, which deep and fully explanation is out of this paper purpose; instead we'll describe its main usage relatively to BEE project.

We used an “Agile Process” template, available in Rational Team Concert, to build BEE project area, after which we customized the process schedule to implement an Agile Model Driven Development (AMDD) [13] process.

After registering team members, we defined process development iteration plan scheduling typical AMDD activities (Envisioning, Iteration Modelling, Model Storming and Development) after an Inception phase. We planned the whole project expecting: first three tools deliverable beginning November 2008, further releases including requirements elicitation, traceability and remote access beginning 2009. Currently Jazz server application represents an important repository storing all project activities carried on so far. This allowed us to greatly simplify project management and to define the best strategies to use to keep its development.

BEE current release is just the first step made toward the final project goal. We decided to rapidly develop a first incomplete release in order to be able to asses both software development effort and process development quality; formal evaluation of either development effort or software size was out of first development iteration purpose. We decided therefore to defer the definition of any measurement system to the next development iteration: now that we are confident with IBM Rational Team Concert reporting features, and have a clearer mind about expected BEE features and development process characteristics, we can set up a measurement system. Such system should be able to assess not only BEE provided functionality, but also development process quality.

Conclusions and future developments

USBD methodology emphasizes the need to elicit software requirements straight from a formal definition of business processes that need to be supported by the software itself: business process modelling for requirements elicitation is therefore an innovative and strategic area of software development methodologies and software tools supporting them. An extensive research in languages to define business processes and existing applications for business process modelling formed the basis for the development of a new business process ontology; such ontology is particularly suitable to realize a software application that proposes itself as an innovative open source tool in the strategic field of business process modelling for requirements elicitation.

Current version of Business process Eclipse Editor allows graphical modelling of business processes, and modelling the business environment surrounding them. In the future, BEE project plans the development of features realizing requirements elicitation and traceability between business processes models and requirements of software supporting those processes. Software requirements will be defined by mean of UML artifacts, namely Use-Cases and Use-Case Realizations; these artifacts will be produced in form of XMI [14] files that can be later imported in common UML modelling software and CASE tools. Use-case models will be derived from business process models as prescribed by USBD methodology; BEE users can flag process participants (`Human` class instances) by setting the `ToBeAutomated` attribute to be true: these performers identify the software to be developed, and activities they

perform identify use-cases; other (not to be automated) performers of such activities identify use-cases actors. Use-case realizations can be derived for non-atomic activities (SubProcesses) performed by `ToBeAutomated` performers together with other performers: the business process model corresponding to such a SubProcess will be used to produce a sequence diagram definition, where performers will be mapped to actors and actions to actors' methods. Traceability will be implemented by means of a *traceability matrix* that identifies correspondences between activities and use-cases.

From a more technical point of view, we foresee a major improvement of BEE thanks to the integration between model and graphical editor at first, as well as the development of more graphical editors based on the same metamodel but intended to allow working on different portions of it, and then the integration of all editors into a single multipage editor; subsequently, requirements elicitation and traceability features will be developed. At the end of the process, BEE will be a multipage editor that integrates one model editor and several graphical editors, allowing to model business processes and business context surrounding them in a simple and intuitive manner, as well as enabling requirements elicitation for the software supporting the business processes. Characteristics of the metamodel we developed also allow possible extensions to the metamodel in order to model a wider range of business process and business context concepts.

References

1. IBM Rational Jazz Community, http://jazz.net/
2. Eclipse Project, http://www.eclipse.org
3. GMF Project, http://www.eclipse.org/modeling/gmf/
4. A. Donatelli, R. Longobardi, R. Gangemi, C. Marinelli: Unified Scenario-Based Design. IBM DeveloperWorks, http://www.ibm.com/developerworks/rational/library/05/1129_donatelli/ (2005)
5. A. Ryan: Beyond Use-Case: Process Driven Development (PDD), http://www.digerateur.com/visualocity/articles/processDrivenDevelopment.jsp (2006)
6. P. Maresca, A. Donatelli, R. Gangemi, R. Longobardi, G. Perrini, R. Correro: Comparing Business Definition Languages. In: First International Conference on Eclipse Technologies Eclipse-IT 2007, pp. 17-26. Cuzzolin Editore (2007)
7. Business Process Modeling Notation (BPMN), http://www.bpmn.org
8. A. Cotugno: Valutazione Comparativa di Applicazioni Open Source per il Business Process Management, http://www.nare.it/uni/beeproject/docs/bpmteva.pdf (2007)
9. Jazz Overview, http://www-01.ibm.com/software/rational/jazz/
10. B. List, B. Korherr: An Evaluation of Conceptual Business Process Modelling Languages, http://wit.tuwien.ac.at/people/list/publications/acmsac2006.pdf (2006)
11. EMF Project, http://www.eclipse.org/modeling/emf/
12. GEF Project, http://www.eclipse.org/gef/
13. Agile Model Driven Development, http://www.agilemodeling.com/essays/amdd.htm
14. XMI, Wikipedia page, http://en.wikipedia.org/wiki/XMI