# Modelling and Execution of Complex Semantic Transactions using WERIGO Metamodel

Alexander Tarabrin

König-Wilhelm-Straße 8, 89073 Ulm
atarabrin@yahoo.com

**Abstract:** Rapid evolution of modern Information Systems and Process-Aware Information Systems (PAIS) in particular requires transactional execution of business logic. The paper introduces the WERIGO Metamodel for modelling of complex transactional behaviour and providing the execution functionality.

## 1 Introduction

Modern information systems require definitely much more functionality the ACID transactions can deliver. Several extended transaction models [1, 2, 3, 4, 5] have been developed to close this gap. The important problem cases are:

- Integration of heterogeneous systems. The classical „Travel Agency Sample" requires transactional execution between several systems. The well known 2PC protocol [6] cannot be applied due to the system heterogeneity and due to long running operations;
- The need of more flexibility. Once an action has been executed, the rollback process can be different depending on diverse factors (state of other actions, time left since the action has been executed etc). Besides it is in some cases desirable to rollback an already committed action.

Semantic Rollback is a known concept to be used in business transactions. The WERIGO Metamodel supports the semantic rollback with the high level of complexity. The structure of this paper is the following: the section 2 introduces the main constructs and the composition rules of the metamodel; the section 3 describes the execution rules the executional functionality is based on; in the section 4 we explain the application of the WERIGO Metamodel using a simple sample.

## 2 Concept

The WERIGO Metamodel was influenced through the ADEPT project [7, 8]. ADEPT is a metamodel for modelling and execution of business processes based on the process definition as a well formed block structured graph. The nodes represent the process activities. The edges represent dependences of different semantics between them (control edges, data edges, sync edges etc).

The Project WERIGO offers a powerful metamodel for modelling of complex transactional dependencies and introduces the mechanisms for transactional execution. The metamodel is based on semantic rollback. The dependency of the compensation method on diverse factors is implemented using the concept of spheres. The term “Sphere” was introduced in several variations [9, 10, 11]. In our metamodel we understand the sphere as a set of activities having the following properties: each sphere element has a predefined rollback action; failure of one of the sphere elements initiates the rollback of other sphere elements been executed successfully; the successful execution of all sphere elements changes the semantic of the rollback action: instead of separate rollback actions of sphere elements a single cumulative rollback activity is used; a sphere element is either a singe activity or an another sphere.

The main constructs of the WERIGO Metamodel are (Fig. 1): four node types (Start Node, End Node, Activity Node, Sphere Node) and two edge types (Commit-Abort Edge and Rollback Edge). The double structure of nodes describes the couple “activity ↔ rollback activity”. The Activity Node (Fig. 1c) consists of an activity (A) and its rollback activity (-A). The rollback activity (-S) of the Sphere Node (Fig. 1d) should be executed instead of the execution of separate rollback activities of the sphere elements.

The nodes are connected together using two types of edges: Commit-Abort Edges (CA-Edges) (Fig. 1e) and Rollback-Edges (R-Edges) (Fig. 1f) for forward and backward execution respectively. The number of incoming and outgoing edges for a node depends on the type of the node and the type and the direction of edges (Table 1).

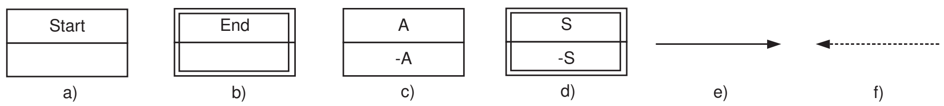


Figure 1. Main constructs of the WERIGO Metamodel

Table 1. Number of incoming and outgoing edges for different node types

| | Commit-Abort-Edges (CA-Edges) | Rollback-Edges (R-Edges) |
|---|---|---|
| **Start Node** | N outgoing | N incoming |
| **End Node** | N incoming | N outgoing |
| **Activity Node** | 1 incoming, 1 outgoing | 1 incoming, 1 outgoing |
| **Sphere Node** | N incoming, 1 outgoing | 1 incoming, N outgoing |

The model is a graph constructed on the following rules: a model has exactly one Start Node and exactly one End Node. The Start Node is connected to all Activity Nodes. Under connection we understand the opposite directed CA-Edge and R-Edge between two nodes. An Activity Node can be connected to a Sphere Node. A Sphere Node can be connected to an another (superordinate) Sphere Node. Sphere Nodes and Activity Nodes having no superordinate Sphere Nodes are connected to the End Node.

## 3 Execution Rules

An activity has a set of predefined states (i.e. Ready, Running, Finished, Failed etc) changing in a known sequence. We describe the states of Nodes using the following statechart diagrams (Fig. 2-4). The Edges (Fig. 1e,f) can be signalled as True or False during the execution process.


Figure 2. Statechart Diagrams for Start Node (a) and End Node (b)

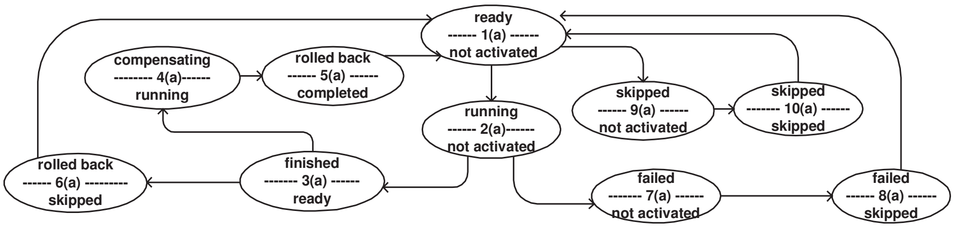


Figure 3. Statechart Diagram for Activity Node

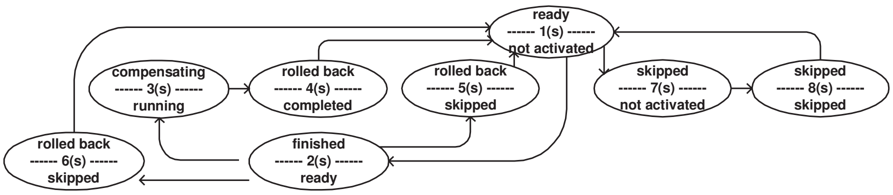


Figure 4. Statechart Diagram for Sphere Node

The execution semantics of the WERIGO Metamodel is the following:

(1) Initially all nodes have the states 1(st), 1(a), 1(s) or 1(e) depending of the node type, the edges are not signalled.
(2) The execution begins on changing the state of the Start Node to 2(st). This initiates the signalling of all outgoing CA-Edges as True.
(3) Signalling of the incoming CA-Edge of an Activity Node as True changes its state to 2(a) – the activity is being executed. Successful executed activity changes the node state to 3(a); the latter initiates signalling of the outgoing CA-Edge as True. A failure or unsuccessful execution of the activity turns the node to the state 7(a). In this case the outgoing CA-Edge is to be signalled as False.
(4) Marking of all incoming CA-Edges of a Sphere Node as True changes the node state to 2(s). This signals its outgoing CA-Edge as True.

(5) Signalling of one of the incoming CA-Edges of a Sphere Node as False initiates the rollback operation for elements of this sphere. The outgoing R-Edges are to be signalled as following: the outgoing R-Edge directed to the failed activity is to be signalled as False. Other outgoing R-Edges should be signalled as True.
(6) Signalling of the incoming R-Edge of an Activity Node as True initiates the execution of the rollback activity (states 4(a) and 5(a)). The latter signals the outgoing R-Edge as False.
(7) Signalling of the incoming R-Edge of an Activity Node as False orders to skip the rollback activity. The Activity Node turns to the state 6(a) (the previous state 3(a)) or to the state 8(a) (the previous state 7(a)). The outgoing R-Edge is to be signalled as False.
(8) Signalling of the incoming R-Edge of a Sphere Node as True initiates the execution of the cumulative rollback activity of the sphere (states 3(s), 4(s)). The latter signals outgoing R-Edges as False. If the rollback activity does not exist, the Sphere Node will be turned to 6(s) and the outgoing R-Edges will be marked as True.
(9) Signalling of the incoming R-Edge of a Sphere Node as False turns the Sphere Node to 6(s) and signals the outgoing R-Edges as False.
(10) Signalling of the all incoming CA-Edges of the End Node as True changes its state to 2(e). The transaction is considered to be committed. The signalling of one of the incoming CA-Edges as False is to be resolved similar to the Rule (5).
(11) Signalling of the all incoming R-Edges of the Start Node changes its state to 3(st). The transaction is rolled back completely.

## 4 Example

The arranging the trip to the conference for a scientist consists of the following steps: conference registration, hotel and flight bookings. Each step has an appropriate rollback step. The booking steps are to be done by a travel agency. After the travel agency has finished its work, the rollback is possible only upon payment of the cancellation fee. We model this sample in WERIGO Metamodel (Fig. 5).

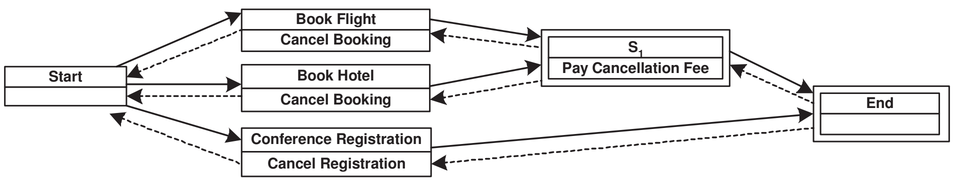


Figure 5. Implementation of the Sample using WERIGO Metamodel

The Start Node turns to the state 2(st) and signals its outgoing CA-Edges as True. The latter turns the Activity Nodes to the state 2(a). After the work has been executed, the Activity Nodes enter the state 3(a) and signal the outgoing CA-Edges as True. The Sphere $S_1$ registers all incoming CA-Edges signal True and turns to 2(s). The outgoing CA-Edge of the Sphere Node signals True. The End Node registers all incoming CA-Edges signal True and turns to 2(e). The execution of the transaction is completed.

Let us review the rollback process for the case, when the participation in the conference could not be registered. The Activity Node „Register Conference“ turns to 6(a); its outgoing CA-Edge signals False. The End Node uses the Rule (5) (see section 3): the R-Edge to the Sphere $S_1$ will be signalled as True, the R-Edge to the failed activity signals False. The Sphere Node $S_1$ was in 2(s), as it registers its incoming R-Edge signals True. The rollback activity „Pay Cancellation Fee“ will be executed. The Sphere Node becomes 3(s) and 4(s) sequentially; the outgoing R-Edges of the Sphere Node will be signalled as False. The latter orders skipping of rollback for the Activity Nodes „Book Flight“ and „Book Hotel“. These two nodes become 6(a) and signal their outgoing R-Edges as False. Parallel to the actions above the Activity Node „Conference Registration“ turns to the state 8(a) and signals its outgoing R-Edge as False. The signalling of all incoming R-Edges turns the Start Node to 3(st). The transaction has been rolled back.

## 5 Conclusion

The WERIGO Metamodel is a singe instrument for modelling and execution of complex transactional dependencies in information systems. The concept can be used either independent or in the cooperation with the PAIS (process-aware information systems). The clear defined execution semantics makes the implementation of the concept easy. In our future work we will introduce the model extensions and investigate the integration of the WERIGO Metamodel in existing PAIS.

## References


[1] Elmagarmid, A.K. (eds): Database Transaction Models for Advanced Applications. Morgan Kaufmann 1992

[2] Hsu, M. (eds): Special Issue on Workflow and Extended Transaction Systems. Data Engineering, IEEE Computer Society technical committee, Vol. 16, No. 2, June, 1993

[3] Jajodia, S.; Kerschberg, L. (Eds.): Advanced Transaction Models and Architectures. Kluwer Academic Publishers, 1997

[4] Leymann, F.; Roller, D.: Production Workflow - Concepts and Techniques. Prentice Hall PTR, 2000

[5] Grefen, P.: Transactional Workflows or Workflow Transactions? DEXA 2002: pp. 60-69

[6] Gray, J.; Reuter, A.: Transaction Processing: Concepts and Techniques. Morgan Kaufmann Publ., 1993

[7] Reichert, M.; Dadam, P.: ADEPTflex - Supporting Dynamic Changes of Workflows Without Losing Control. Journal of IIS, Kluwer Academic Publ., Vol. 10, No. 2, pp. 93-129

[8] Reichert, M. et al.: Adaptive Process Management with ADEPT2. Proc. Int'l Conf. on Data Engineering, ICDE 2005, Tokyo, April 2005, Demo Session

[9] Davies C.T.: Spheres of Control. IBM Systems Journal vol. 17,no. 2. pp 179 –198.

[10] Leymann, F.: Supporting Business Transactions via Partial Recovery in WfMS. Proc. Datenbanksysteme in BTW, Dresden, March 1995, Springer-Verlag, S. 51-70

[11] Hagen, C.J.: A Generic Kernel for Reliable Process Support. Dissertation, ETH Zürich, 1999