

Software Engineering within a Digital Business Ecosystem

Giulio Marcon

Salzburg University of Applied Sciences, A-5412 Puch / Salzburg, Austria



July 6, 2006

Authors

- Giulio Marcon

Salzburg University of Applied Sciences, A-5412 Puch / Salzburg, Austria

- Angelo Corallo

eBMS - ISUFI, University of Lecce, 73100 Lecce, Italy

- Maurizio de Tommasi

eBMS - ISUFI, University of Lecce, 73100 Lecce, Italy

- Thomas Heistracher

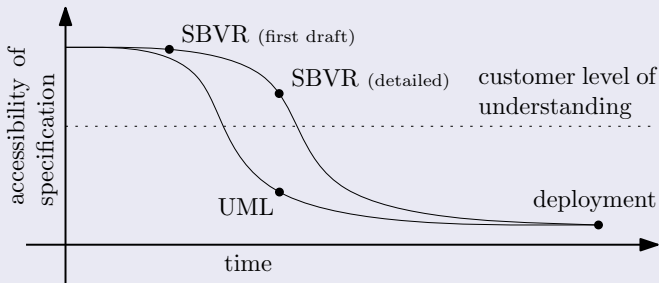
Salzburg University of Applied Sciences, A-5412 Puch / Salzburg, Austria



Outline

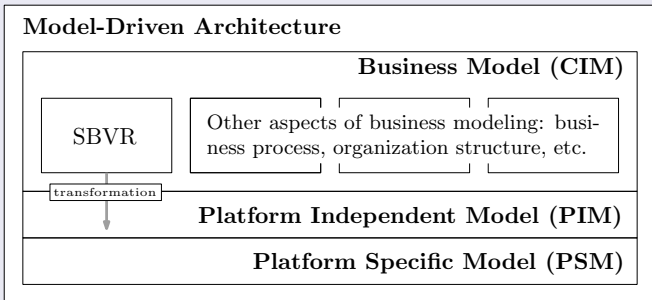
- 1 Natural language software specification
- 2 An ecosystem of companies and components
- 3 Code generation
- 4 Synchronization of specification from adapted software

Natural language software specification



- Standard current practices (UML) drop very soon below the level of understanding of the customers, contrarily to natural language based techniques (SBVR).

SBVR in the Model-Driven Architecture



- According to the vision of the Object Management Group, SBVR is the language of choice for Computational Independent Models in the Model-Driven Architecture.

SBVR model example

branch

Concept Type: organization function

Definition: rental organization unit *that has rental responsibility*

car movement

Definition: planned movement of a rental car of a specified car group from a sending branch to a receiving branch

receiving branch

Concept Type: role

Definition: branch *that is the destination of a car movement*

sending branch

Concept Type: role

Definition: branch *that is the origin of a car movement*

car movement has receiving branch

Necessity: *each* car movement *has exactly one* receiving branch

car movement has sending branch

Necessity: *each* car movement *has exactly one* sending branch

rental car is assigned to car movement

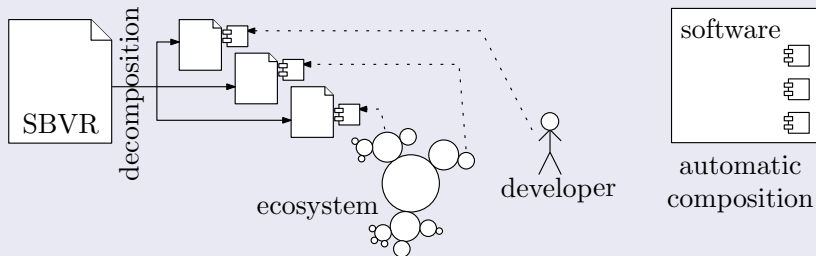
Necessity: *At most one* rental car *is assigned to each* car movement

car movement being international

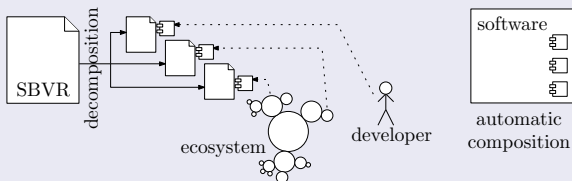
Concept Type: characteristic

Definition: car movement *having country of sending branch that is not the country of receiving branch of the car movement.*

Service Oriented Architecture with a Digital Ecosystem

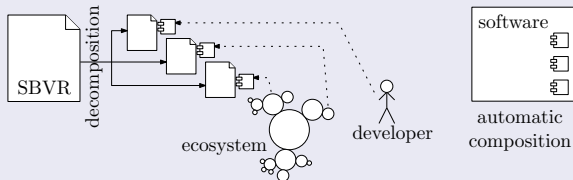


How a request can be satisfied



- Some implementor finds the request and offers to develop it;
- several components existing in the digital ecosystem can be re-used to satisfy the request and the additional parts necessary are developed by some implementors;
- all the necessary components are already in the digital ecosystem and only fine-grained optimization prior to deployment needs to be done.

Levels of automation

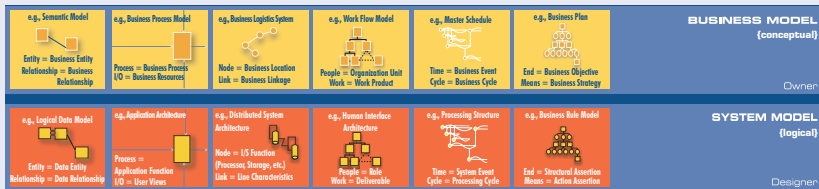


- Decompose the request/specification in atomic units;
- match the atomic components to the ones available;
- offer the best solutions to the requester;
- combine the components to satisfy the request;
- open bids for developers to implement missing components.

Zachman Framework

WHAT	HOW	WHERE	WHO	WHEN	WHY	
DATA	FUNCTION	NETWORK	PEOPLE	TIME	MOTIVATION	
<p>List of Things Important to the Business</p>  <p>Entity = Class of Business Thing</p>	<p>List of Processes the Business Performs</p>  <p>Process = Class of Business Process</p>	<p>List of Locations in Which the Business Operates</p>  <p>Node = Major Business Location</p>	<p>List of Organizations Important to the Business</p>  <p>People = Major Organizational Unit</p>	<p>List of Events/Cycles Significant to the Business</p>  <p>Time = Major Business Event/Cycle</p>	<p>Lists of Business Goals/Strategies</p>  <p>Ends/Means = Major Business Goal/Strategy</p>	<p>SCOPE (contextual)</p> <p>Planner</p>
<p>e.g., Semantic Model</p>  <p>Entity = Business Entity Relationship = Business Relationship</p>	<p>e.g., Business Process Model</p>  <p>Process = Business Process I/O = Business Resources</p>	<p>e.g., Business Logistics System</p>  <p>Node = Business Location Link = Business Linkage</p>	<p>e.g., Work Flow Model</p>  <p>People = Organization Unit Work = Work Product</p>	<p>e.g., Master Schedule</p>  <p>Time = Business Event Cycle = Business Cycle</p>	<p>e.g., Business Plan</p>  <p>End = Business Objective Means = Business Strategy</p>	<p>BUSINESS MODEL (conceptual)</p> <p>Owner</p>
<p>e.g., Logical Data Model</p>  <p>Entity = Data Entity Relationship = Data Relationship</p>	<p>e.g., Application Architecture</p>  <p>Process = Application Function I/O = User Views</p>	<p>e.g., Distributed System Architecture</p>  <p>Node = I/S Function (Process/Storage, etc.) Link = Line Characteristics</p>	<p>e.g., Human Interface Architecture</p>  <p>People = Role Work = Deliverable</p>	<p>e.g., Processing Structure</p>  <p>Time = System Event Cycle = Processing Cycle</p>	<p>e.g., Business Rule Model</p>  <p>End = Structural Assertion Means = Action Assertion</p>	<p>SYSTEM MODEL (logical)</p> <p>Designer</p>
<p>e.g., Physical Data Model</p>  <p>Entity = Segment/Table/etc. Relationship = Pointer/Key/etc.</p>	<p>e.g., System Design</p>  <p>Process = Computer Function I/O = Data Elements/Sets</p>	<p>e.g., Technology Architecture</p>  <p>Node = HW/System Software Link = Line Specifications</p>	<p>e.g., Presentation Architecture</p>  <p>People = User Work = Screen Formats</p>	<p>e.g., Control Structure</p>  <p>Time = Execute Cycle = Component Cycle</p>	<p>e.g., Rule Design</p>  <p>End = Condition Means = Action</p>	<p>TECHNOLOGY MODEL (physical)</p> <p>Builder</p>
<p>e.g., Data Definition</p>  <p>Entity = Field Relationship = Address</p>	<p>e.g., Program</p>  <p>Process = Language Statement I/O = Control Block</p>	<p>e.g., Network Architecture</p>  <p>Node = Address Link = Protocol</p>	<p>e.g., Security Architecture</p>  <p>People = Identity Work = Job</p>	<p>e.g., Timing Definition</p>  <p>Time = Interrupt Cycle = Machine Cycle</p>	<p>e.g., Rule Specification</p>  <p>End = Sub-condition Means = Step</p>	<p>DETAILED REPRESENTATIONS (out-of-context)</p> <p>Subcontractor</p>

In Zachman Framework terms



- From row 2, the conceptual business model, to row 3, the logical system model.

From SBVR models to UML class diagrams and code

car movement being international

Concept Type: *characteristic*

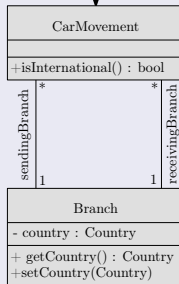
Definition: *car movement having country of sending branch that is not the country of receiving branch of the car movement.*

Code Generation

UML Class Diagram generation

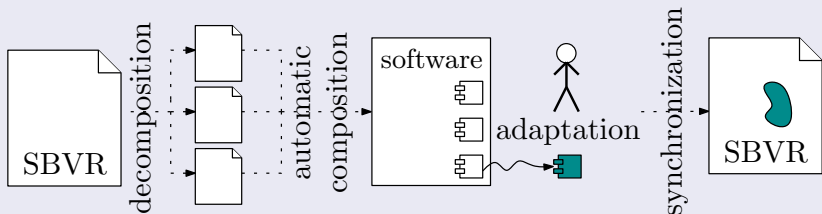
```
class CarMovement {
    private Branch sendingBranch;
    private Branch receivingBranch;

    /* SBVR_Concept_Type: Characteristic */
    public bool isInternational () {
        Country countryFrom = sendingBranch.getCountry();
        Country countryTo = receivingBranch.getCountry();
        return !countryFrom.equals(countryTo);
    }
}
```



- Example from ruleset for transformation of unary fact types (characteristics).

Re-adaptation of specification



Round-trip engineering with SBVR

- An SBVR model is created;
- the corresponding UML models are generated;
- code is generated from the annotated UML models;
- UML models or the source code are modified;
- the original SBVR model is updated with the modification.

References



Heistracher, T., Kurz, T., Marcon, G., Masuch, C.:

Collaborative software engineering with a digital ecosystem.

In: Proc. International Conference on Global Software Engineering, Costão do Santinho, Florianópolis, Brazil (2006) (accepted).



Nachira, F.:

Toward a network of digital business ecosystems fostering the local development.

Discussion paper,

<http://www.digital-ecosystems.org/doc/discussionpaper.pdf> (2002)



OMG:

Semantics of Business Vocabulary and Business Rules Specification. (2006) First interim specification, <http://www.omg.org/docs/dtc/06-03-02.pdf>.



G. Marcon, H. Okada, T. Kurz, and T. Heistracher.

D16.3, Report on Adaptive Service Generator.

DBE Project, EU-IST 507953, June 2006.

Software Engineering within a Digital Business Ecosystem

Giulio Marcon

Salzburg University of Applied Sciences, A-5412 Puch / Salzburg, Austria



July 6, 2006