Università degli Studi di L'Aquila

Dottorato di Ricerca in Informatica e Applicazioni

XVI Ciclo – 2004, *Defense 2004-06-28*

Anna Rita Laurenzi

An XML Methodology to Model and Use Scenarios in the Software Development Process

Computer Science Department- University of L'Aquila (Italy) laurenzi@di.univaq.it

Scenarios

how the users will be using the system

1. taking a user's point of view, to describe system external behaviour, promote and advance tight user involvement in the system development
2. improve communication between the different group involved in system development
3. user-system interactions are an ideal base to design system test cases
4. provide guidelines to build prototype
5. build a valuable base for project documentation and user manual

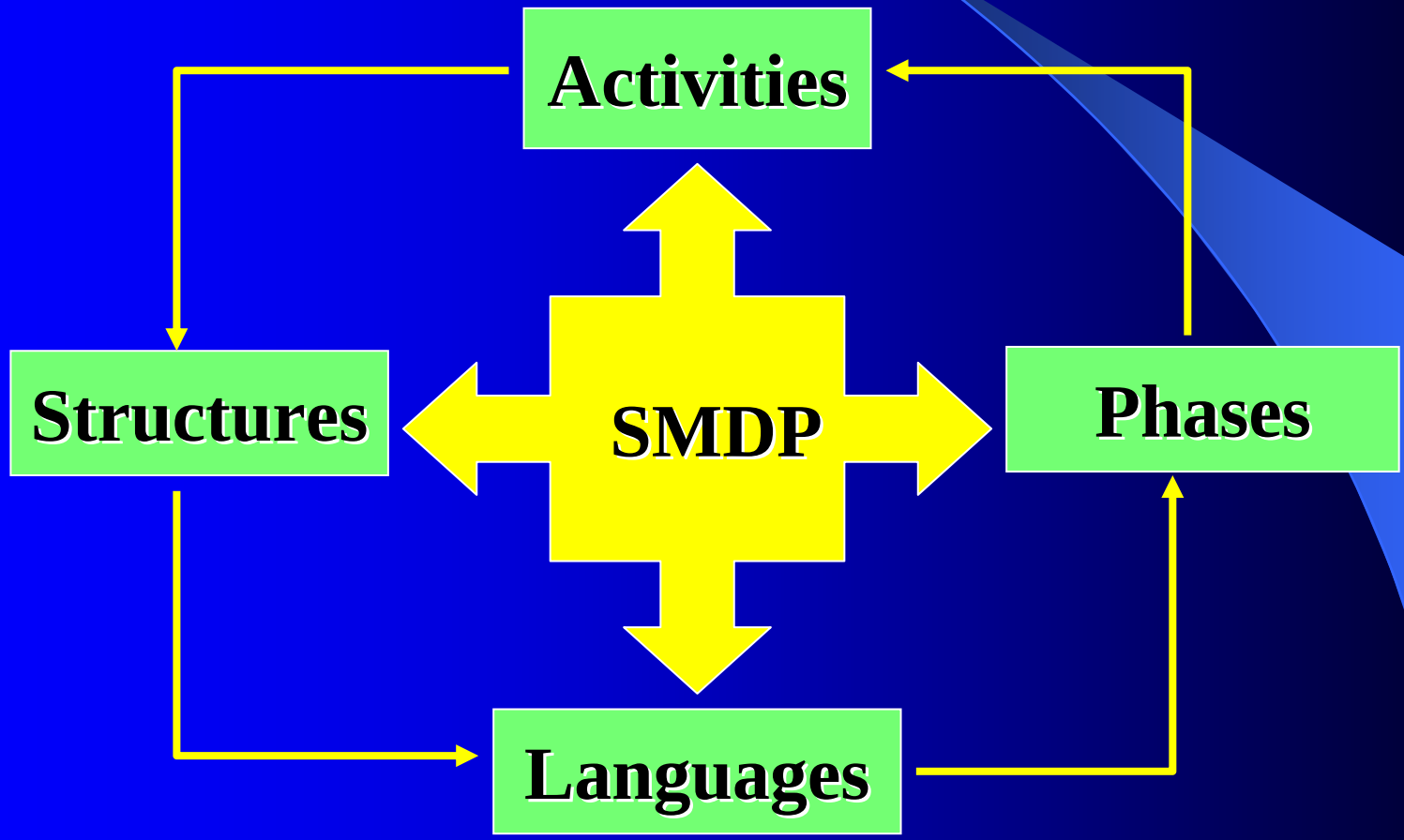
Motivation

- the lack of a unique modelling process and a unique environment, to support the scenario life cycle and the scenario management
- the lack of transformation techniques to reuse scenarios in other phases of SE, e.g. testing phase, design phase
- the lack of a proper structure and semantics for *scenario model*, conceived as a collection of scenarios integrated in a unique model through the composition operators

Thesis Contribution

- the definition of a modelling process to support the life cycle and scenario management, based on:
  - a scenario model that preserves the potential benefits of scenario written using informal language
  - a set of transformation rules to derive other form of specification, i.e. test cases, without introduce intermediate formalism
- the development of a prototype software environment to support the entire methodology
- the application of the process on a wide variety of case studies in order to refine and validate the methodology

Scenario Model Development Process (SMDP)



SMDP Characteristics

- Lightweight
- XML based
- Textual and Graphical
- Multi viewpoint
- Iterative and Incremental

SMDP Structures/Languages

- **Scenario Model**: is the structure for describing a set of scenarios composed through appropriate composition rules
- **Test Model**: is the structure for describing a set of system test cases, establishing that the implemented system conforms to its requirements
- **Behaviour Model**: is the structure for describing a set of state diagrams, representing the dynamic behaviour of the system.

X M L

SDML

ScenarioDescription MarkupLanguage

TDML

Test case Description MarkupLanguage

BDML

State Diagram Description Markup Language

SMDP
Structure/Languages/Presentation System

Scenario Model

Test Model

Behaviour Model

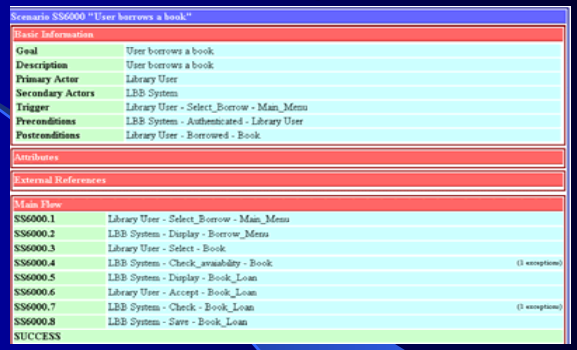
XML

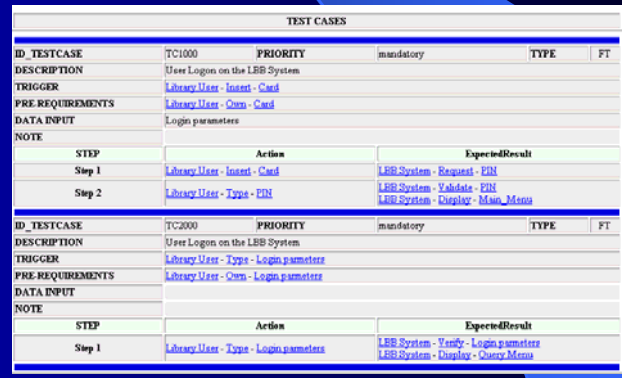
SDML

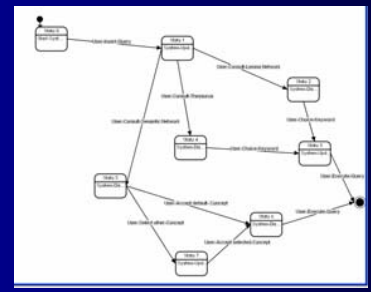
TDML

BDML

XSL



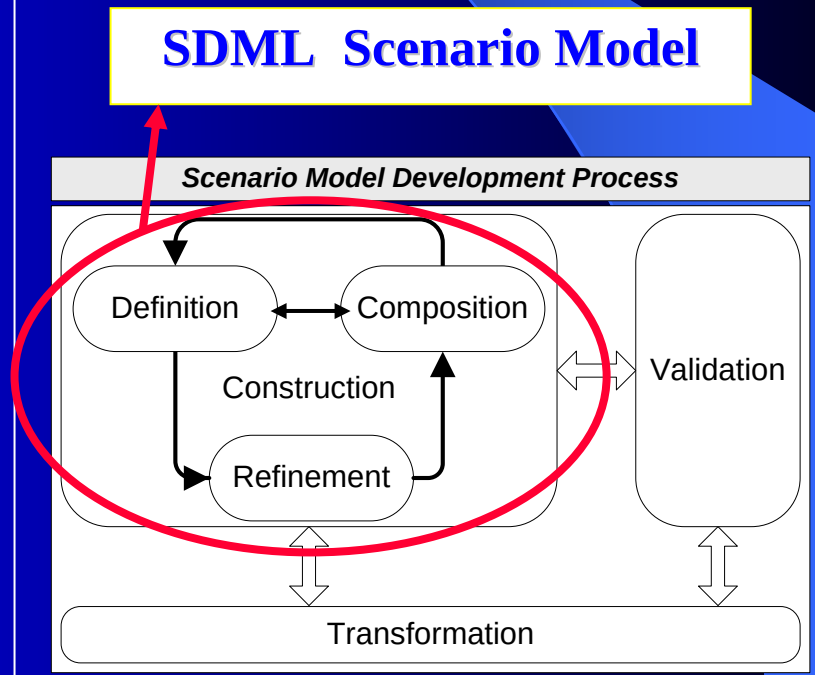




SMDP Phases

- definition (where specifications are described and formalized in the form of scenarios)
- refinement (the elaboration of more detailed scenarios)
- composition (the integration phase, where different scenarios are composed)
- transformation (the derivation phase, where other forms of specifications are derived from scenarios)

- validation (the quality attributes for scenarios are checked)

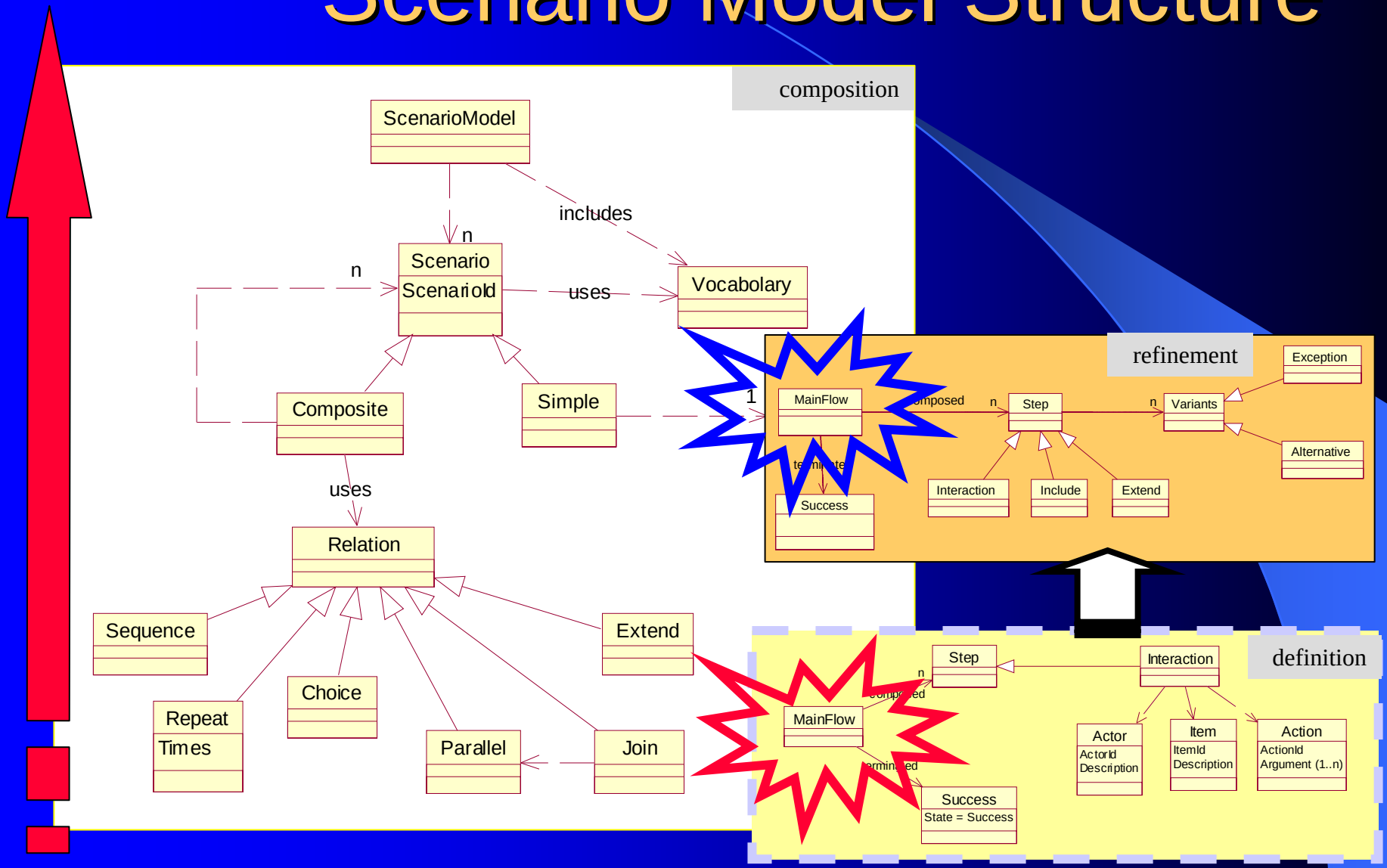


Basic Terminology

- ***Interaction***: a triple (subject, action, object). Which take place between user and system.
- ***Goal***: the result expected from the execution of the user-system interactions
- ***Main flow***: describes a sequence of interactions that allows to achieve the primary actor's goal.

- ***Exception Flow***: a sequence of interactions that represent obstacles to the primary actor's goal achievement
- ***Alternative Flow***: an alternative sequence of interactions that lead to success of the primary actor's goal.
- ***Scenario***: a collection of possible flow of interactions, showing how the primary actor's goal might be delivered or fail.

Scenario Model Structure



An example of Composition Rule

Let S1 and S2 be scenarios.
If precondition(S2)=postcondition(S1) and primaryActor(S1)= primaryActor(S2), then S = **Sequence** (S1, S2) is a scenario such that:

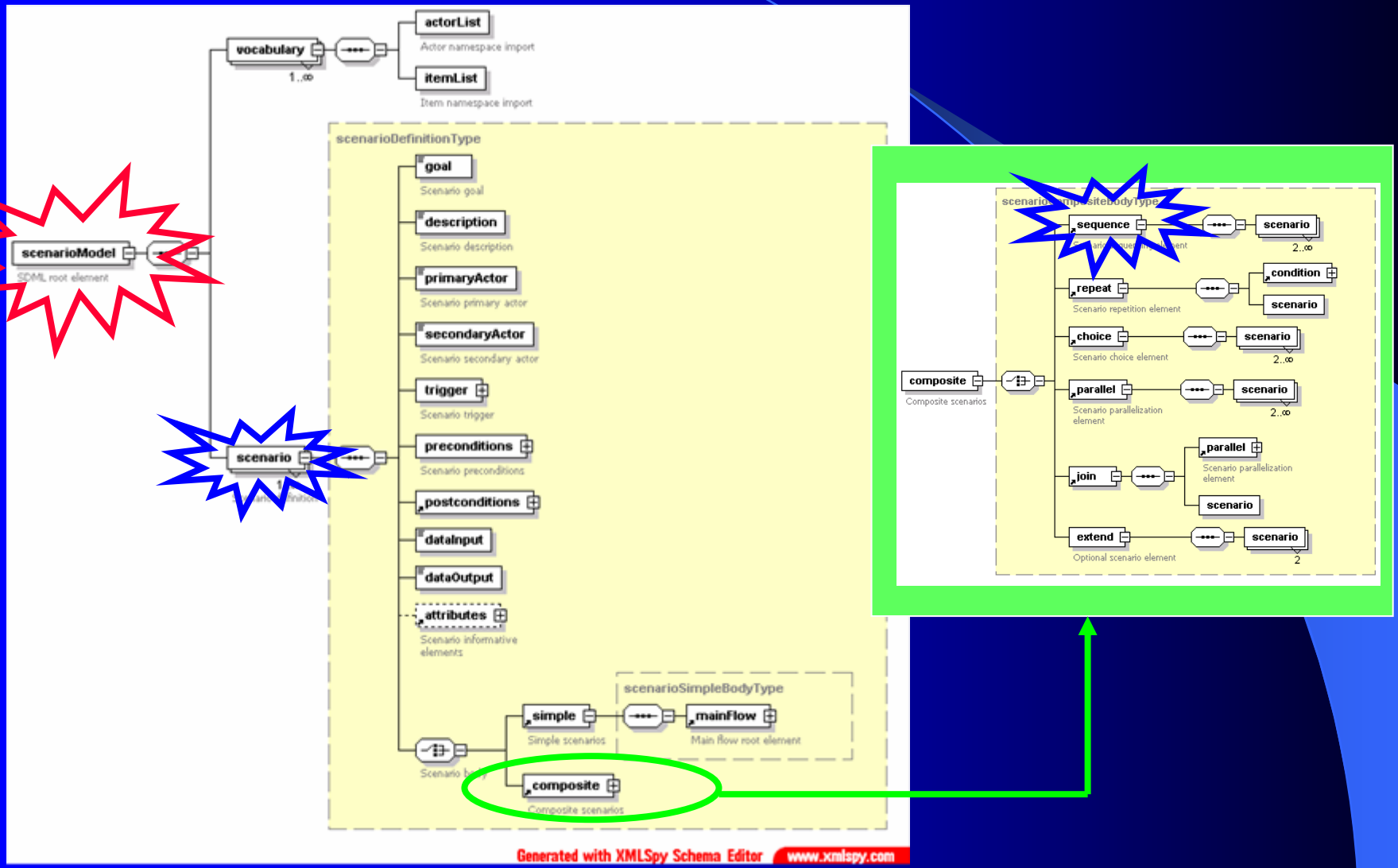
- goal(S)= (goal(S1), goal(S2))
- primaryActor(S)=primaryActor(S1)
- trigger(S)=trigger(S1)
- precondition(S)=precondition(S1)
- postcondition(S)=postcondition(S2)

flows of interactions (S) are obtained combining the flows of interactions(S1) and flows of interactions(S2)

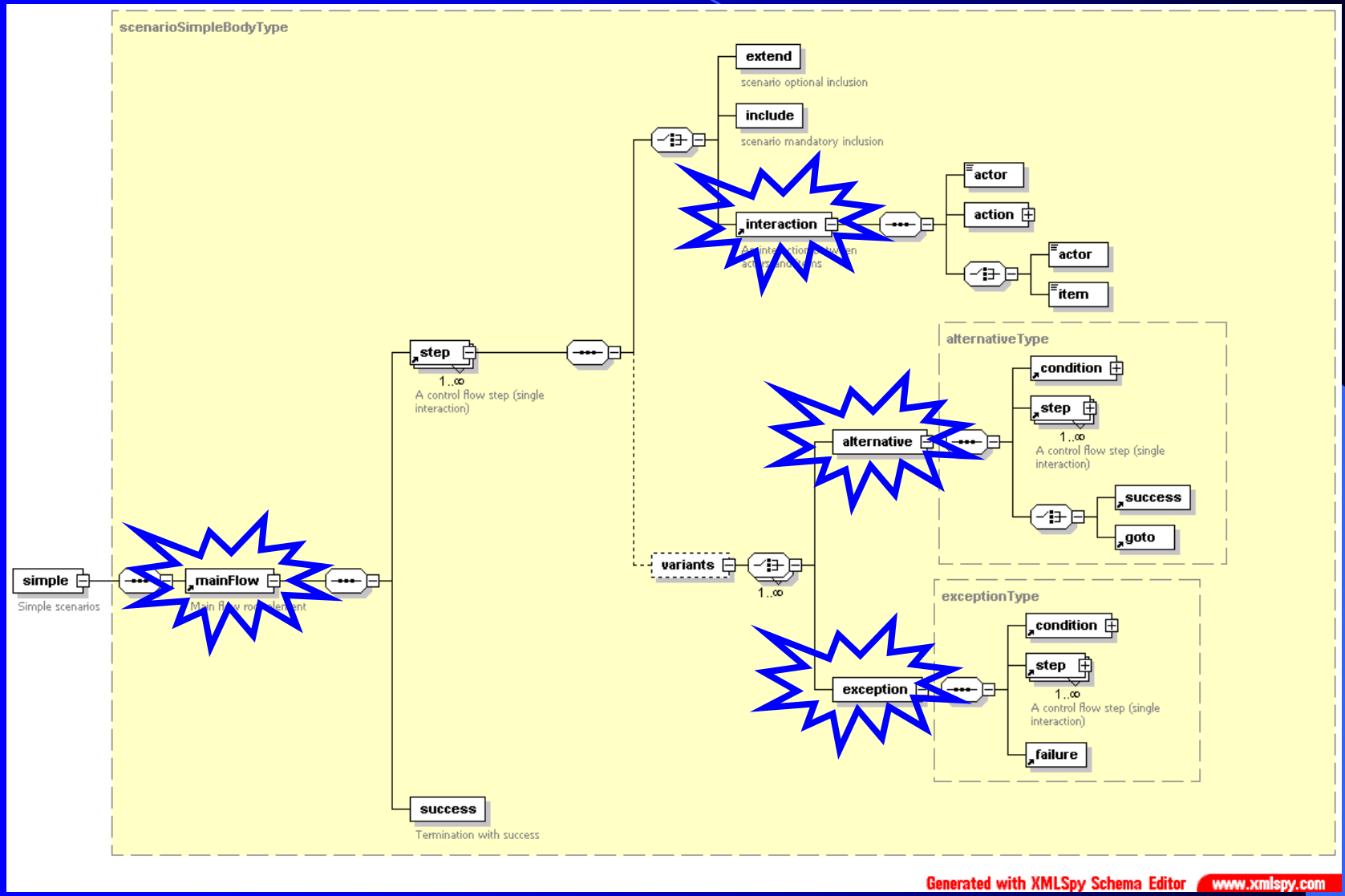
Scenario Model Structure

- it provides a complete specification of a system's behaviour, providing great expressiveness and unambiguous interaction semantics
- it helps avoid the explosion of scenarios that occur if we try to simply list all the possibile flow of interaction in undisciplined manner
- it allows to integrate scenarios directly without introduce other formalism, improving the quality of requirements specification and keeping scenarios persistent

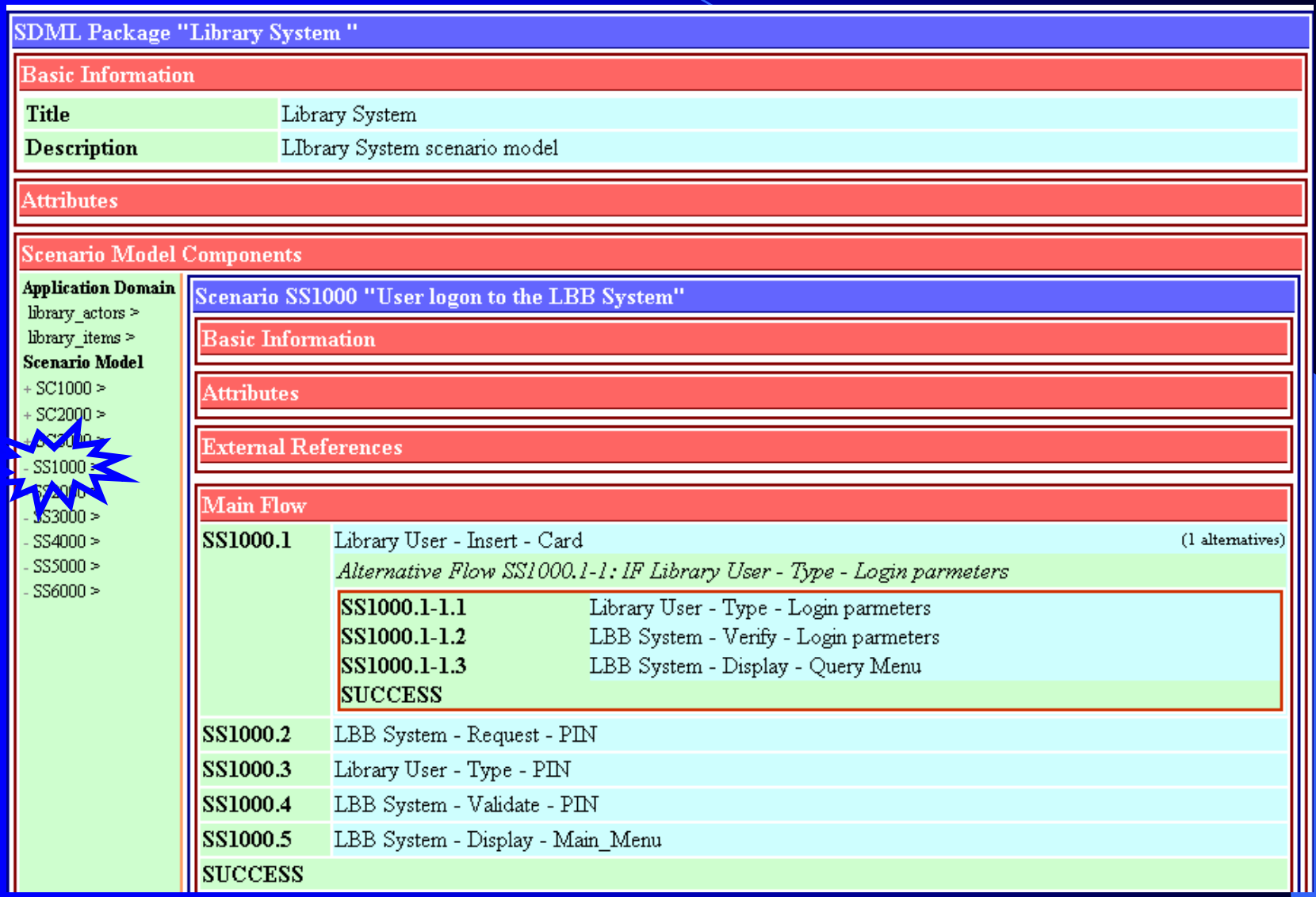
SDML Scenario Model



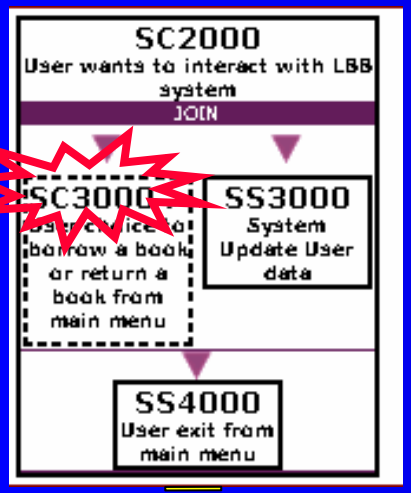
SDML Scenario Model

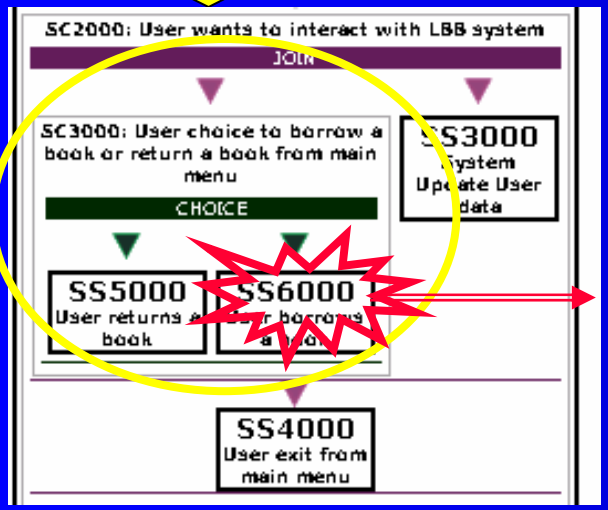


Scenario Model Presentation



Scenario Model Presentation





Scenario SS6000: "User borrows a book"

Basic Information	
Goal	User borrows a book
Description	User borrows a book
Primary Actor	Library User
Secondary Actors	LBB System
Trigger	Library User - Select_Borrow - Main_Menu
Preconditions	LBB System - Authenticated - Library User
Postconditions	Library User - Borrowed - Book

Attributes

External References

Main Flow		
SS6000.1	Library User - Select_Borrow - Main_Menu	
SS6000.2	LBB System - Display - Borrow_Menu	
SS6000.3	Library User - Select - Book	
SS6000.4	LBB System - Check_avaiability - Book	(1 exceptions)
SS6000.5	LBB System - Display - Book_Loan	
SS6000.6	Library User - Accept - Book_Loan	
SS6000.7	LBB System - Check - Book_Loan	(1 exceptions)
SS6000.8	LBB System - Save - Book_Loan	
SUCCESS		

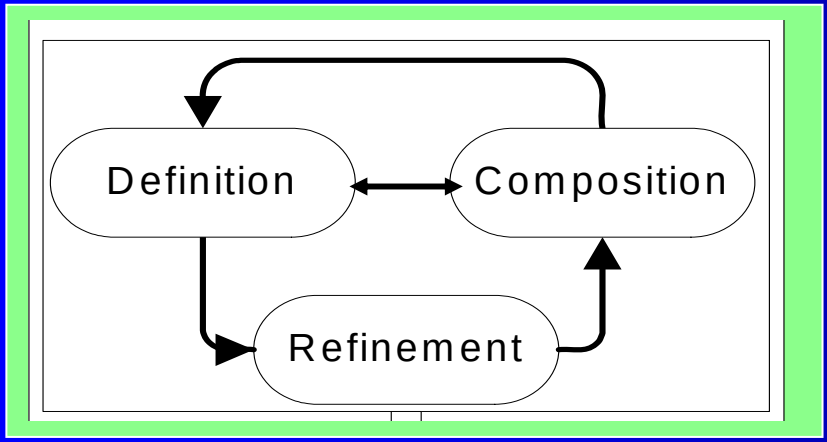
Scenario Model Presentation

Scenario SS6000 "User borrows a book"	
Basic Information	
Attributes	
External References	
Main Flow	
SS6000.1	Library User - Select_Borrow - Main_Menu
SS6000.2	LBB System - Display - Borrow_Menu
SS6000.3	Library User - Select - Book
SS6000.4	LBB System - Check_avaiability - Book
	Exception Flow SS6000.4-1: IF (LBB System - Check_avaiability - Book = false)
	SS6000.4-1.1 LBB System - Eject - Card
	SS6000.4-1.2 LBB System - Display - ErrorMessage
	FAIL
SS6000.5	LBB System - Display - Book_Loan
SS6000.6	Library User - Accept - Book_Loan
SS6000.7	LBB System - Check - Book_Loan
	Exception Flow SS6000.7-1: IF (Book.reserved > 5)
SS6000.8	LBB System - Save - Book_Loan
SUCCESS	

(1 exceptions)

(1 exceptions)

Construction Phase Activities



Composition Phase

composition of different scenarios through integration rules

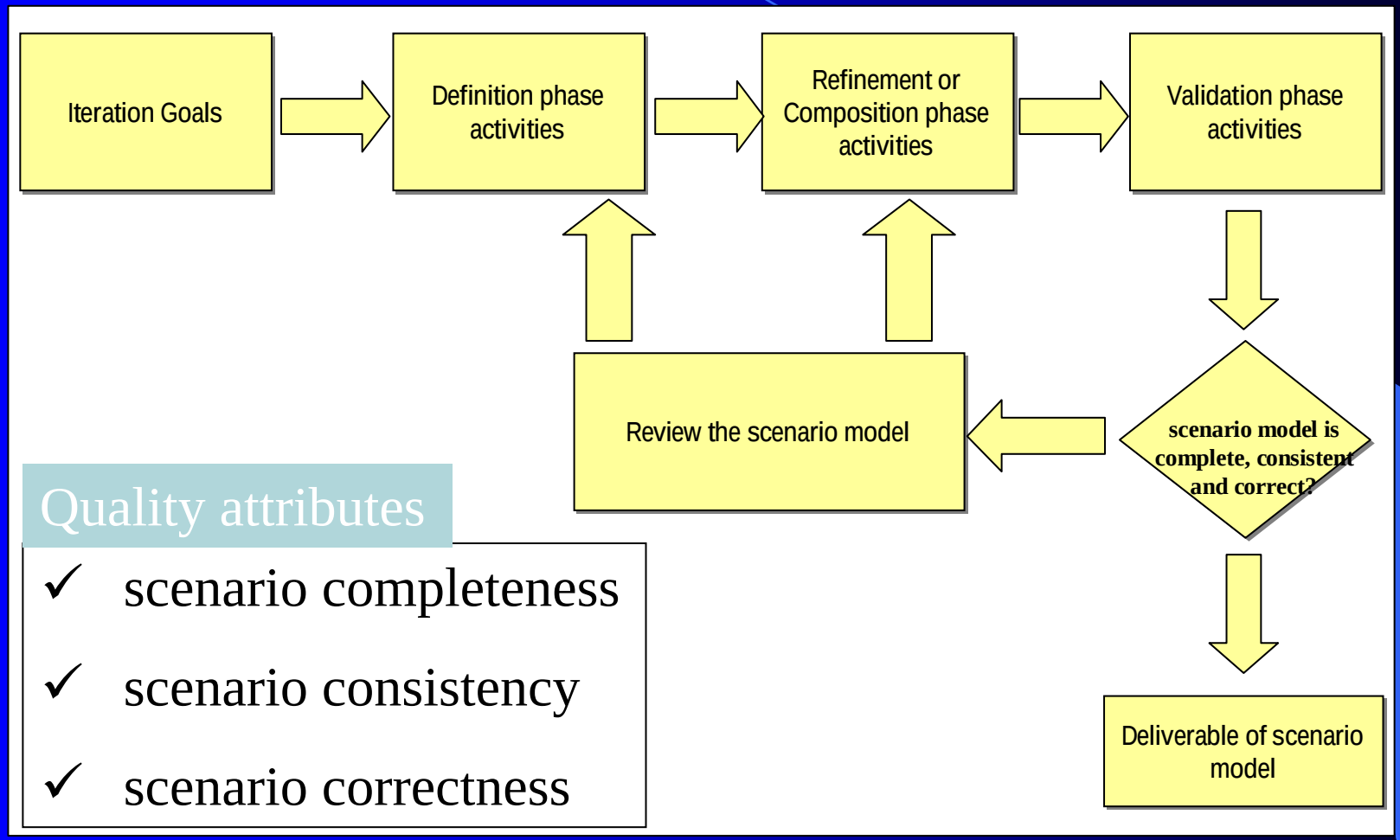
Definition Phase

- the definition of Vocabulary: list of actors, items, actions
- the identification of the scenario's goals, and the basic list of triggers-condition, pre-condition.
- the description of a mainflow of interaction for each identified goal

Refinement Phase

- the identification of alternatives flows
- the identification of exceptions flows (goal obstacles)
- the identification of redundant flows (include)
- the identification of extensional flows(extend)

Validation Phase



Quality attributes

- ✓ scenario completeness
- ✓ scenario consistency
- ✓ scenario correctness

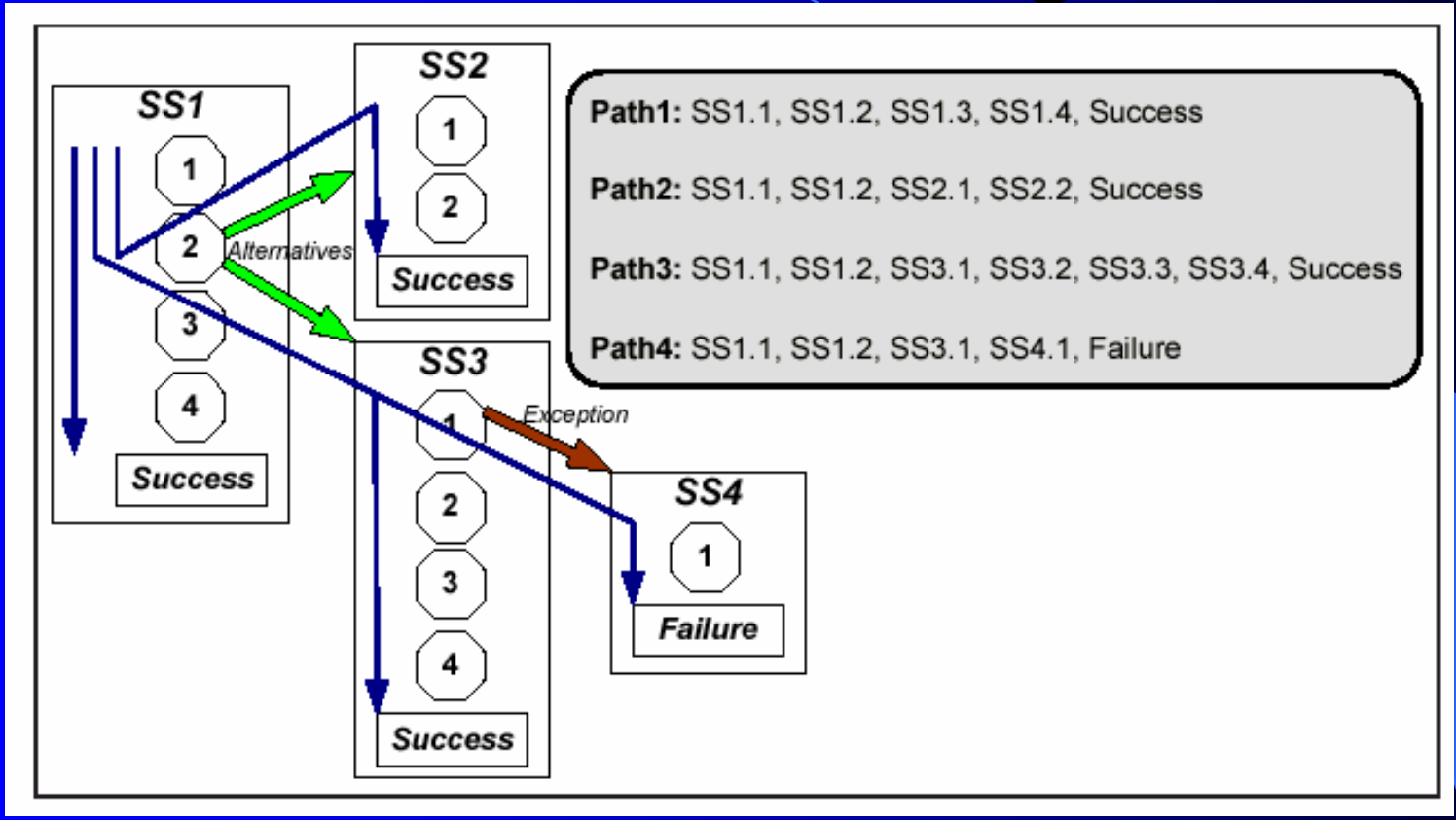
Validation Phase

- **Scenario Completeness**: the capability of scenarios to represent all the behaviour required.
  - iterative and incremental characteristics of SMDP
  - variant, extend, include, composition rules
  - indicator (i.e. number of: steps of scenarios,variants)
- **Scenario Consistency**: the capability to avoid contradictions inside the scenario model and respect its derived models
  - vocabulary shared between models
  - cross- referencing obtained through XML
  - variant related to the same goal
- **Scenario Correctness**: the capability to avoid incorrecteness during flow of interactions composition

Trasformation Phase

- to derive other forms of specifications :
  - to support validation phase, reducing incompleteness and inconsistency for the scenario model during the construction phase
  - to support other phases of software development, such as testing phase and design phase

Path Generation Algorithm



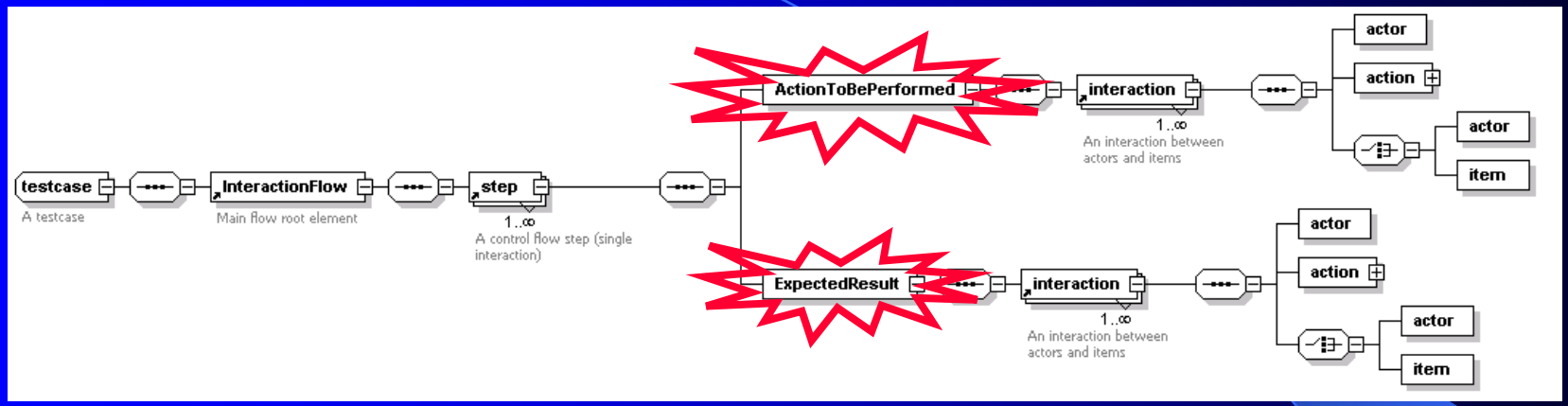
A path is defined as an interaction flow included in a scenario, and it corresponds to a flow of actions obtained by analysing all the flows of interaction derived from that scenario (simple or composite).

Path Generation Algorithm

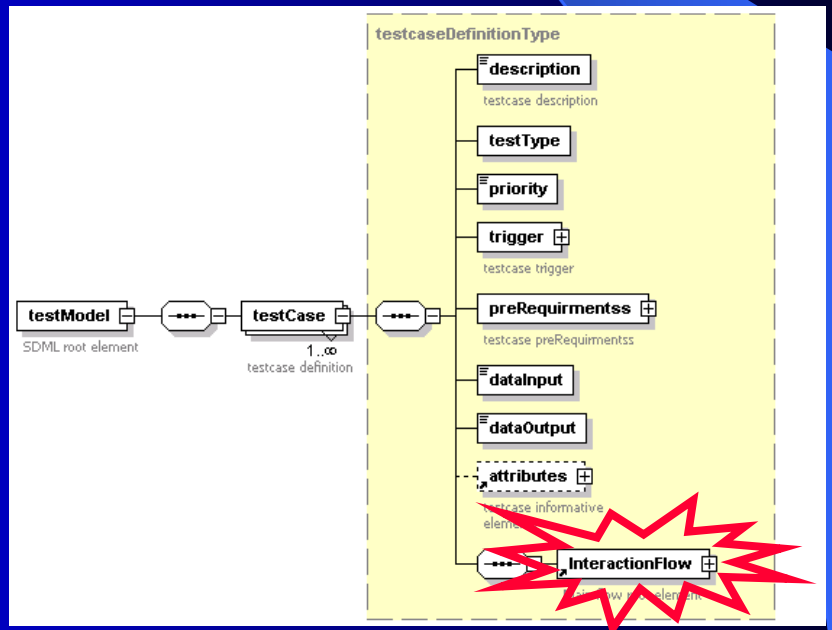
```
path_step st {
/*prolog denotes a set of interactions common to a set of subflows */
prolog Ø;
while (st does not contain variants and st <> success or failure){
  if (st is an interaction) {
   append st to prolog ;
    let st' be the step following st in the current interaction flow ;
   set st to st';
  } else if (st is a goto) {
      while {st is a goto) {
         set st to the step reached by the goto in the current interaction flow;
       }//while
    }// else if
}//while

if (st contains variants) {
  let st' be the step following st in the current interaction flow ;
  foreach st''in variants(st ) {
    /*subflows(x) is the set of interaction flows starting from the step x */
    set subflows (st'') to path_step(st'');
  }//foreach
  /*mainsubflows is the set of interaction flows that follow the step st */
  set mainsubflows to path_step(st')
  foreach st''in variants(st) {
    insert prolog at the beginning of each interaction flow contained in subflows [st''];
  }//foreach
  insert prolog then step at the beginning of each interaction flow contained in
     mainsubflows ;
  return the union of subflows and mainsubflows ;
  } else {
     return prolog ;
   }// else
}
```

Test Model Generation



1. ActionToBePerformed: scenario interactions that have user as primary actor
2. ExepectedResult: scenario interactions that have system as primary actor

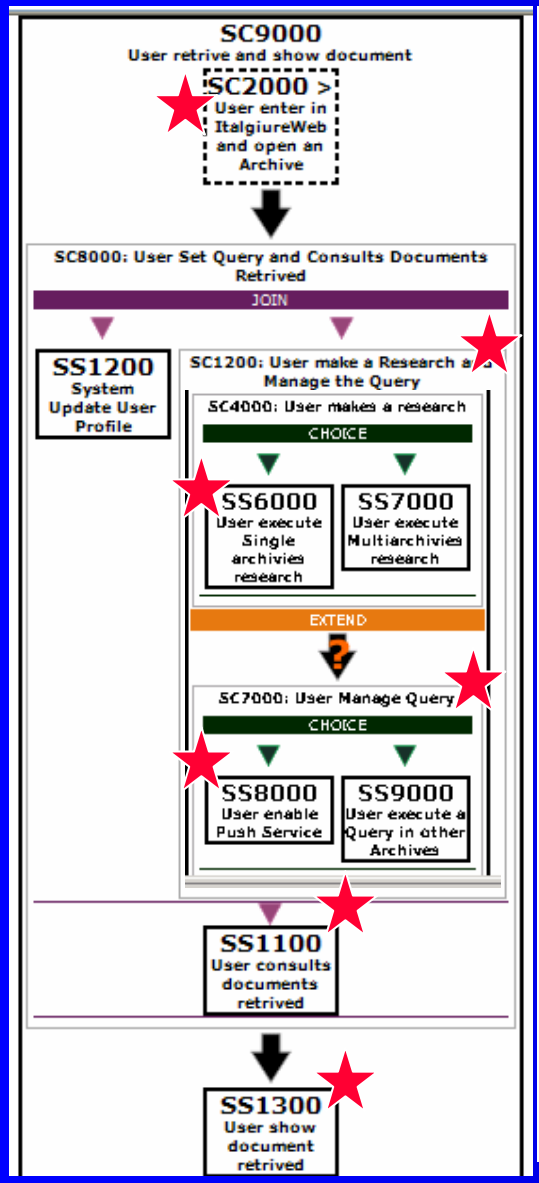


Main Flow		
SS1000.1	Library User - Insert - Card	
	Alternative Flow SS1000.1-1: IF Library User - Type - Login parmeters	
	SS1000.1-1.1	Library User - Type - Login parmeters
	SS1000.1-1.2	LBB System - Verify - Login parmeters
	SS1000.1-1.3	LBB System - Display - Query Menu
	SUCCESS	
SS1000.2	LBB System - Request - PIN	
SS1000.3	Library User - Type - PIN	
SS1000.4	LBB System - Validate - PIN	
SS1000.5	LBB System - Display - Main_Menu	
SUCCESS		

TEST CASES

ID_TESTCASE	TC1000	PRIORITY	mandatory	TYPE	FT
DESCRIPTION	User Logon on the LBB System				
TRIGGER	Library User - Insert - Card				
PRE-REQUIREMENTS	Library User - Own - Card				
DATA INPUT	Login parameters				
NOTE					
STEP	Action		ExpectedResult		
Step 1	Library User - Insert - Card		LBB System - Request - PIN		
Step 2	Library User - Type - PIN		LBB System - Validate - PIN LBB System - Display - Main_Menu		

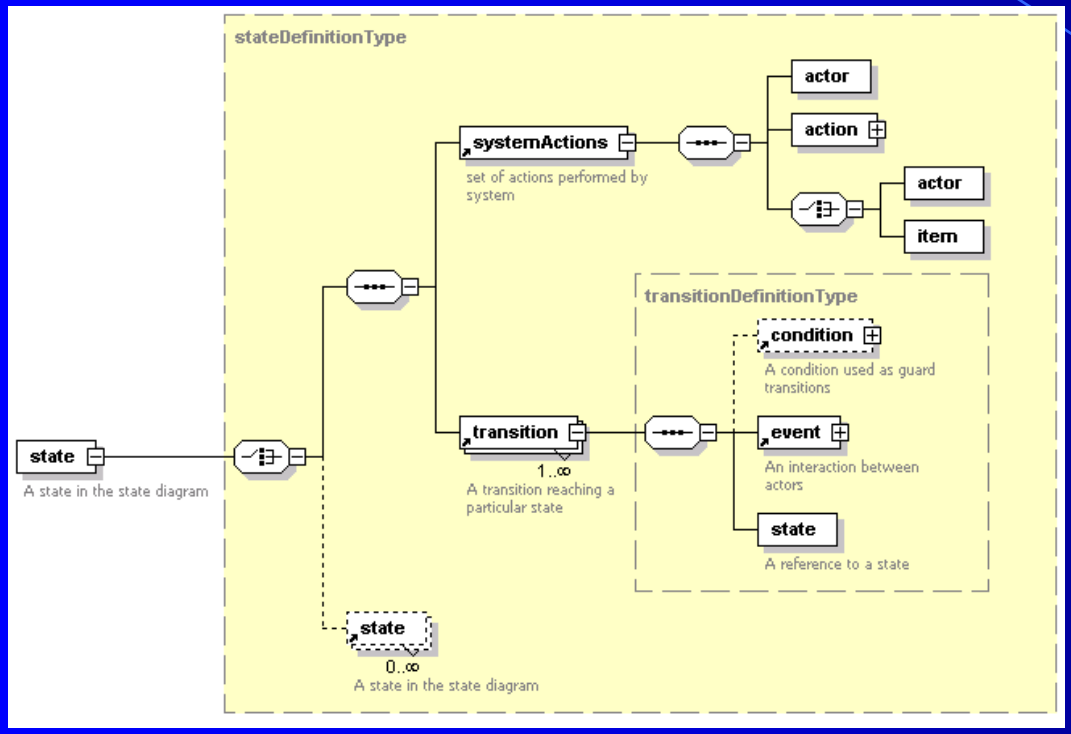
ID_TESTCASE	TC2000	PRIORITY	mandatory	TYPE	FT
DESCRIPTION	User Logon on the LBB System				
TRIGGER	Library User - Type - Login parmeters				
PRE-REQUIREMENTS	Library User - Own - Login parmeters				
DATA INPUT					
NOTE					
STEP	Action		ExpectedResult		
Step 1	Library User - Type - Login parmeters		LBB System - Verify - Login parmeters LBB System - Display - Query Menu		



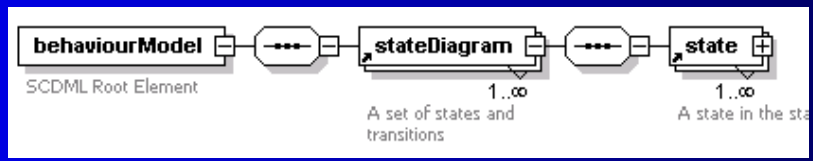
ID_TESTCASE	TC9000	PRIORITY	mandatory	TYPE	FT
DESCRIPTION	User retrive and show document				
TRIGGER	User - Open - Browser IE6.0				
PRE-REQUIREMENTS	User - know - URL				
DATA INPUT	URL=[www.italgiure.giustizia.it] Login parameters=[guest,guest] Archivie Name= [Civile] Query=[testo=minore] Keyword=[minori]				
NOTE					

STEP	Action	ExpectedResult
Step 1	User - Open - Browser IE6.0	System - Load - Browser IE6.0
Step 2	User - insert - URL	System - Display - Home Page
Step 3	User - Select - Archive	System - Request - Login parameters
Step 4	User - Type - Login parameters	System - Verify - Login parameters System - Show Login correct - Message System - Show - Archive
Step 5	User - Insert - Query	System - Update - Query Area
Step 6	User - Consult - Lemma Network	System - Display - Keyword
Step 7	User - Choice - Keyword	System - Update - Query
Step 8	User - Execute - Query	System - Check - Query System - Execute - Query System - Update - Research Tree
Step 9	User - Save - Query	System - Ask_Name - Query
Step 10	User - Insert_Name - Query.	System - Save - Query
Step 11	User - Active - Push Service	System - Enable - Push Service
Step 12	User - Consult - Set of Document	System - Show Titles - Set of Document
Step 13	User - Select - Document	System - Show - Document

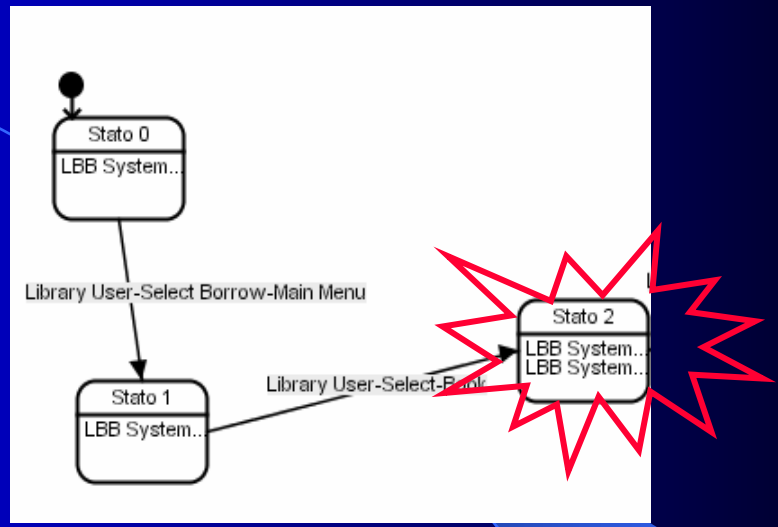
Behaviour Model Generation



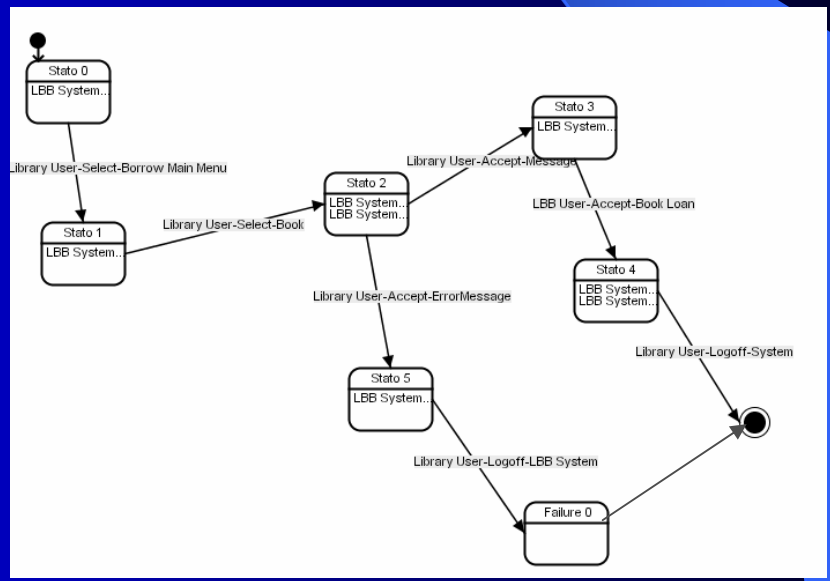
1. system action: scenario interactions that have system as primary actor
2. transition: scenario interactions that have user as primary actor



Main Flow		
SS6000.1	Library User - Select_Borrow - Main_Menu	
SS6000.2	LBB System - Display - Borrow_Menu	
SS6000.3	Library User - Select - Book	
SS6000.4	LBB System - Check_avaiability - Book	(1 exceptions)
	Exception Flow SS6000.4-1 : IF (LBB System - Check_avaiability - Book =false)	
	SS6000.4-1.1 LBB System - Eject - Card **SS6000.4-1.2** LBB System - Display - ErrorMessage **FAIL**	
SS6000.5	LBB System - Display - Book_Loan	
SS6000.6	Library User - Accept - Book_Loan	
SS6000.7	LBB System - Check - Book_Loan	(1 exceptions)
SS6000.8	LBB System - Save - Book_Loan	
SUCCESS		



Main Flow		
SS6000.1	Library User - Select_Borrow - Main_Menu	
SS6000.2	LBB System - Display - Borrow_Menu	
SS6000.3	Library User - Select - Book	
SS6000.4	LBB System - Check_avaiability - Book	(1 exceptions)
	Exception Flow SS6000.4-1 : IF (LBB System - Check_avaiability - Book =false)	
	SS6000.4-1.1 LBB System - Display - Message **SS6000.4-1.2** Library User - Accept - ErrorMessage **SS6000.4-1.3** LBB System - Eject - Card **FAIL**	
SS6000.5	LBB System - Display - Message	
SS6000.6	Library User - Accept - Message	
SS6000.7	LBB System - Display - Book_Loan	
SS6000.8	Library User - Accept - Book_Loan	
SS6000.9	LBB System - Check - Book_Loan	(1 exceptions)
SS6000.10	LBB System - Save - Book_Loan	
SUCCESS		

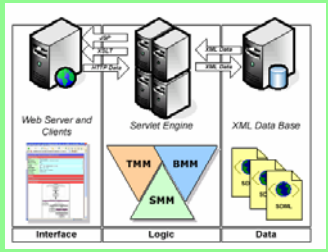


Outline

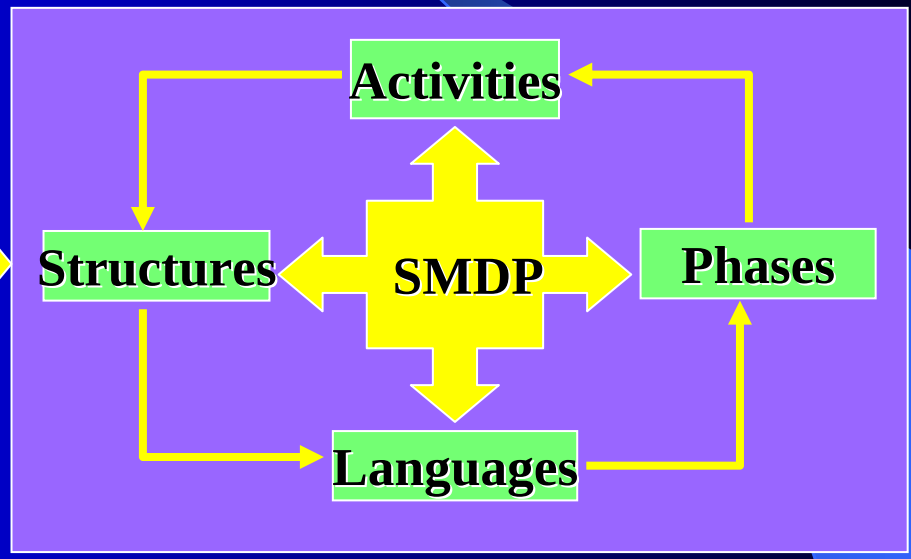
- Scenario Model Development Process
  - ✓ Overview
  - ✓ Scenario Model Structure
  - ✓ Scenario Description Markup Language
  - ✓ Scenario Model Presentation System
  - ✓ Phases and Activities
  - ✓ Test Model and Behaviour Model
- Scenario Model Environment
- Case Study: ItalgiureWeb System
- Conclusion and Future works

Scenario Model Environment

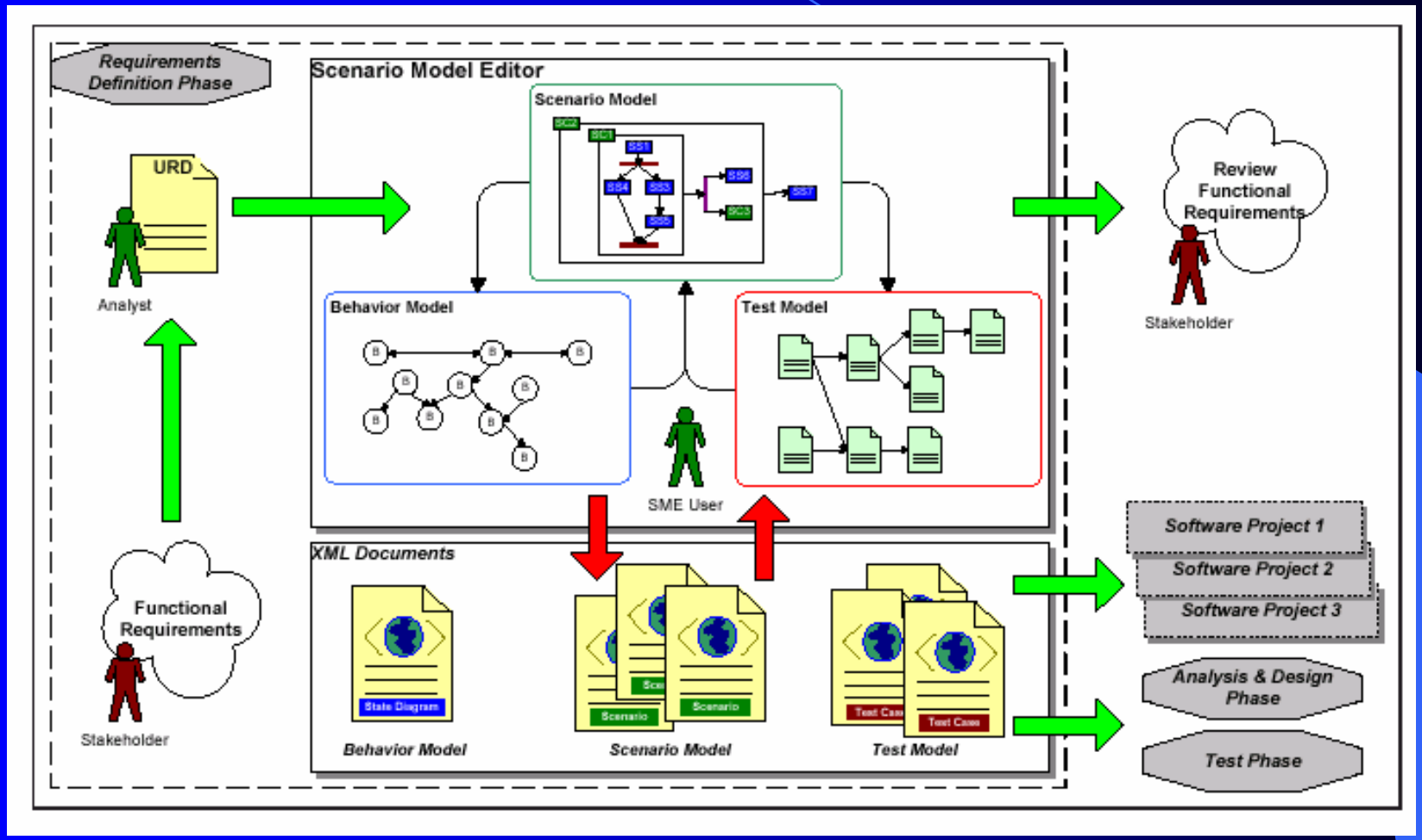
SME



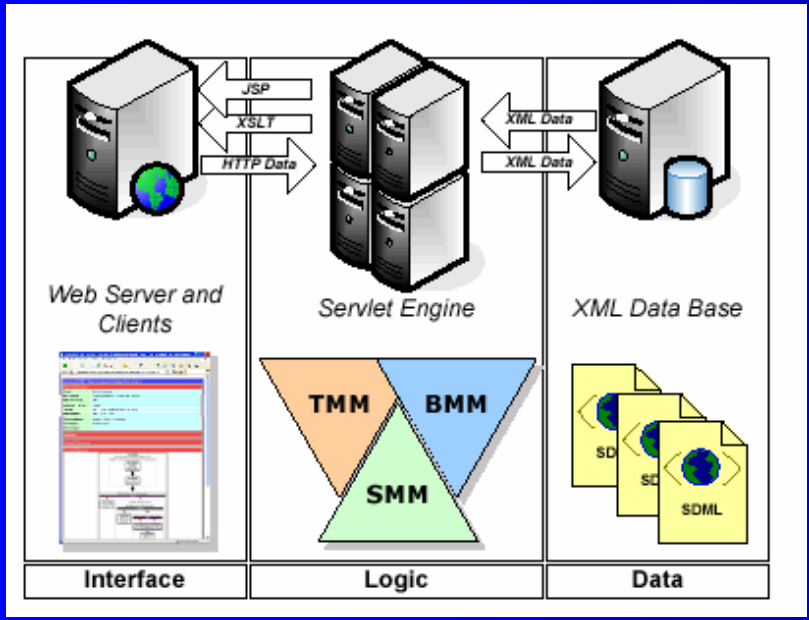
supports



The use of SME



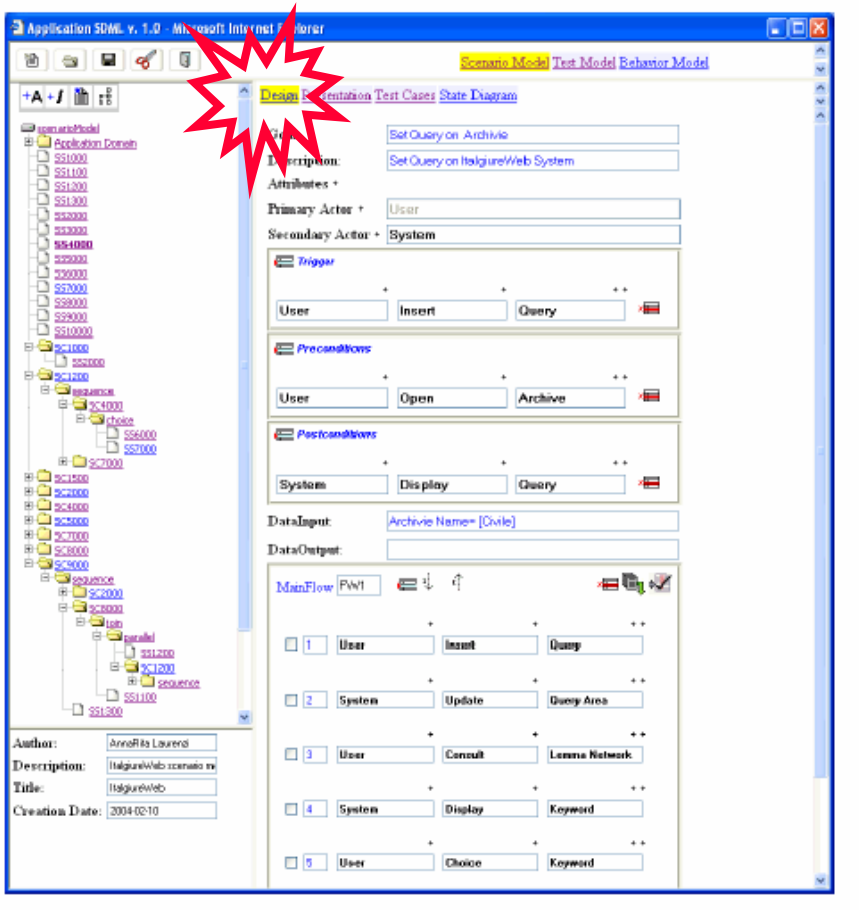
SME→ Architectural Overview



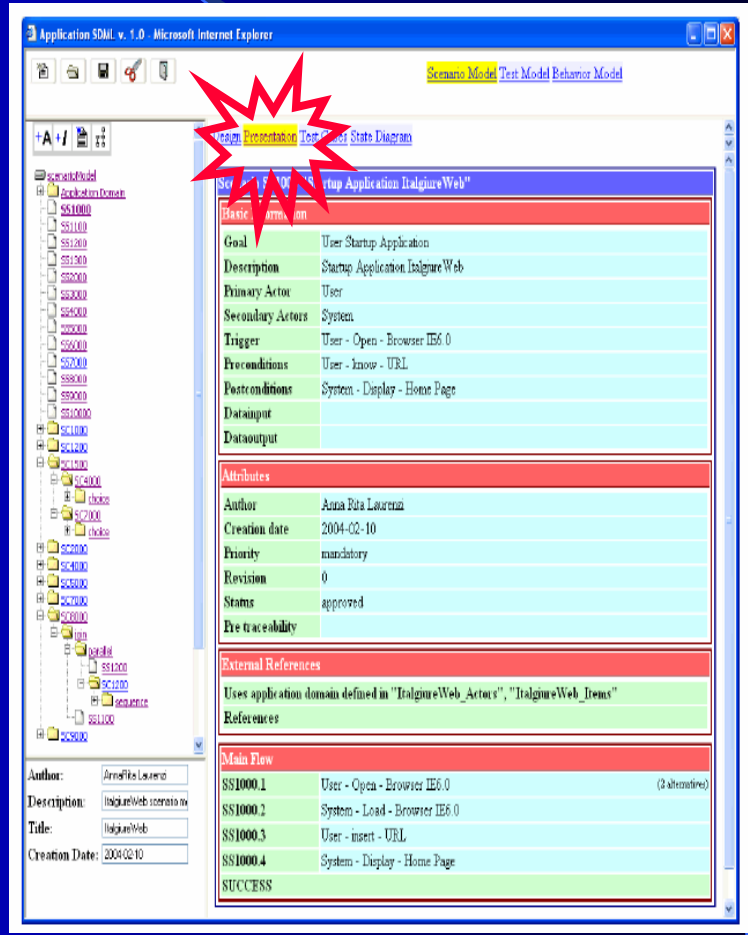
- **Scenario Model Manager (SMM):** supports the creation,maintenance, validation and visualisation of the SDML documents corresponding to the scenarios that compose the scenario model.
- **Test Model Manager (TMM):** allows to automatically derive a set of test cases from the entire scenario model.The test model is visualised in a tree –structured fashion,where each test case has an hypertextual format.
- **Behavior Model Manager (BMM):** allows to encode the entire scenario model into a set of state diagrams.The behaviour model can be visualised through an external viewer.

Scenario Model Manager

Design View

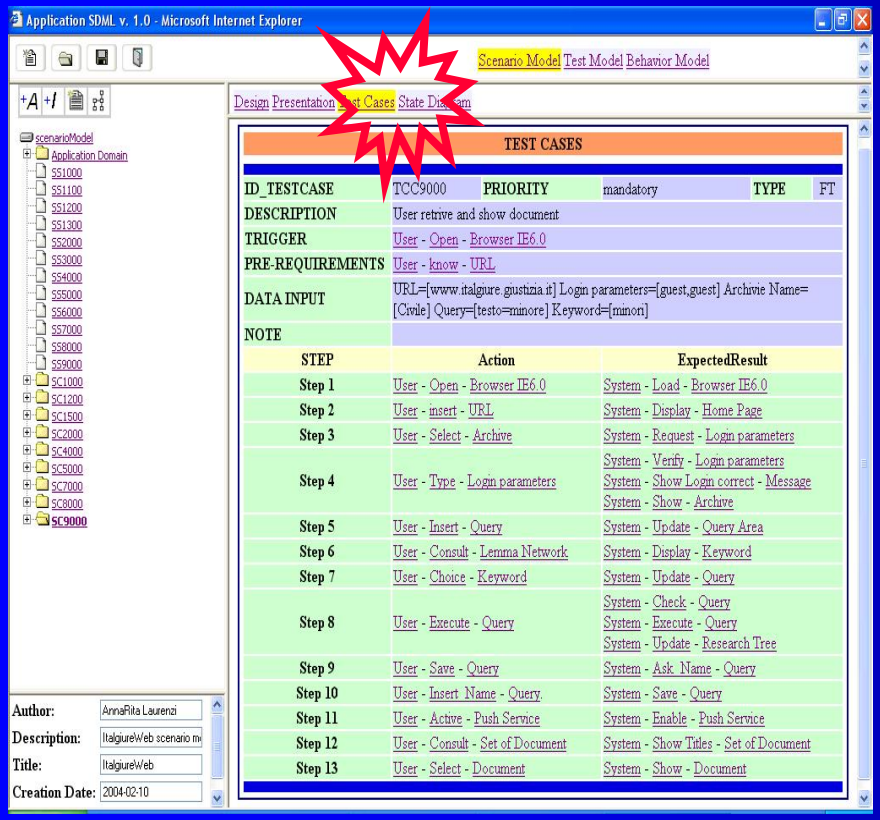


Presentation View

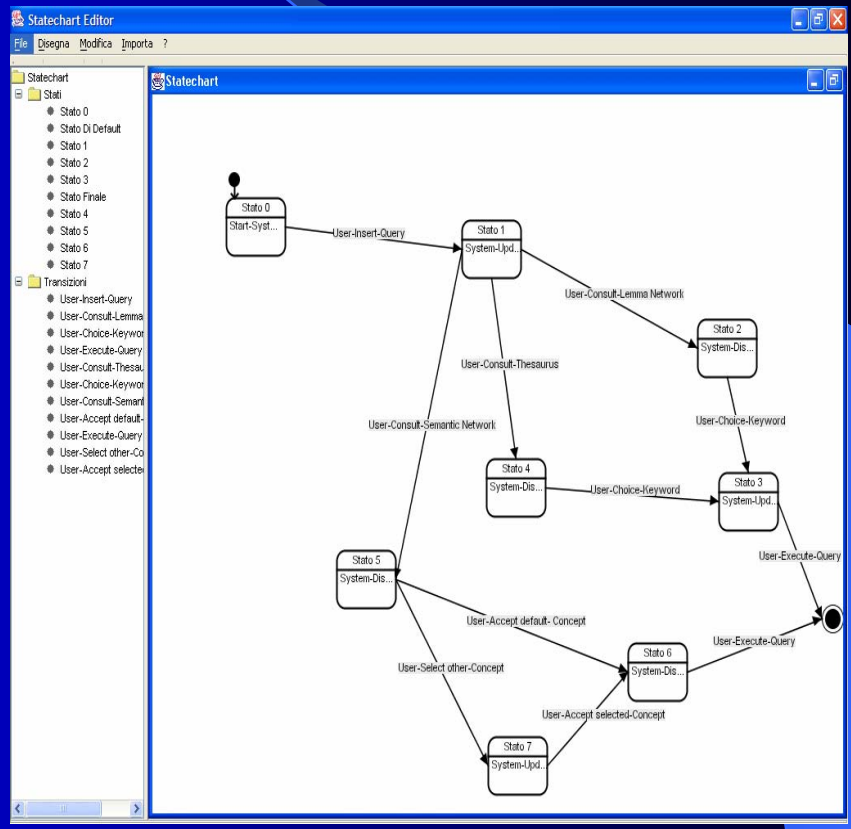


Scenario Model Manager

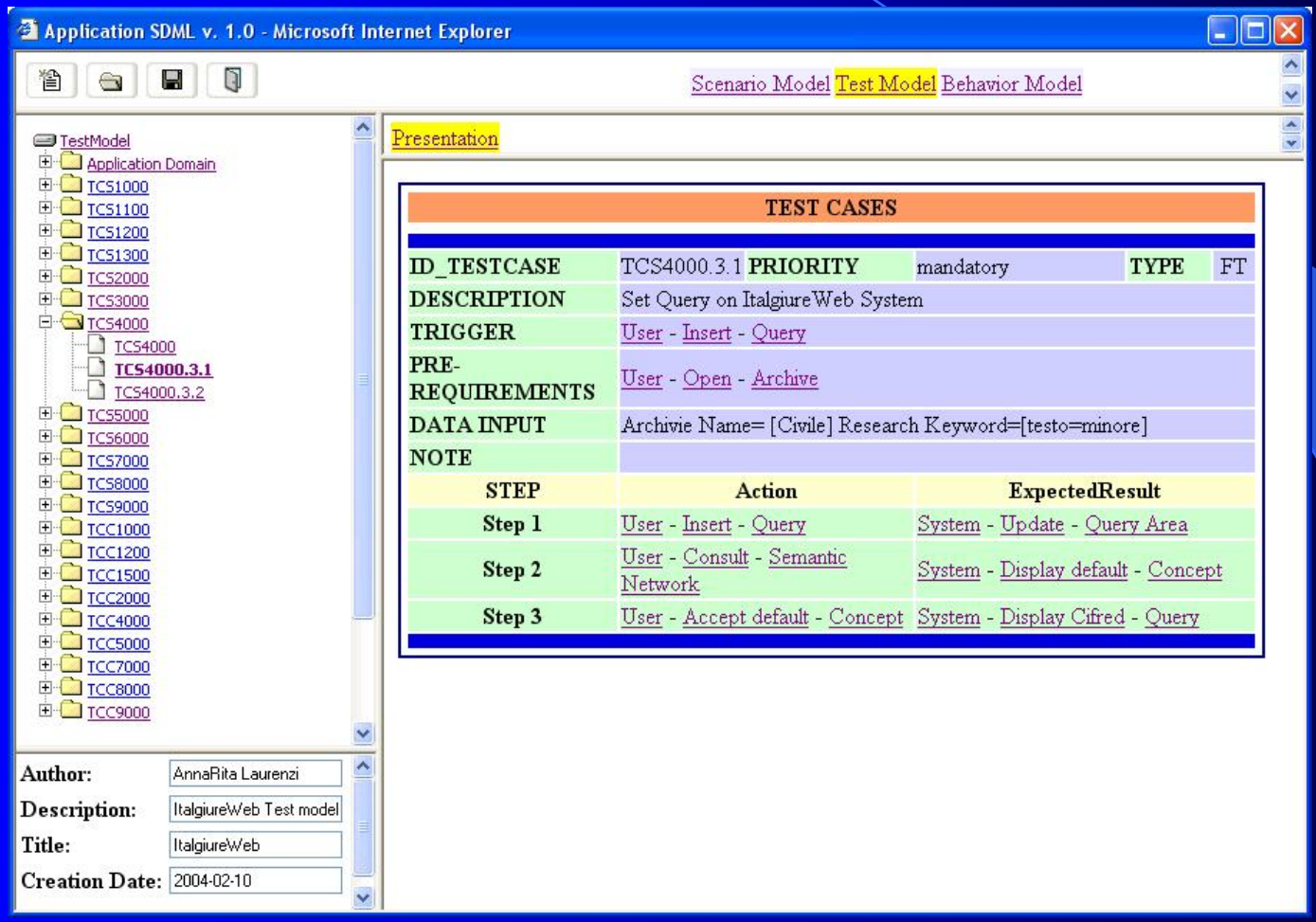
Test Cases View



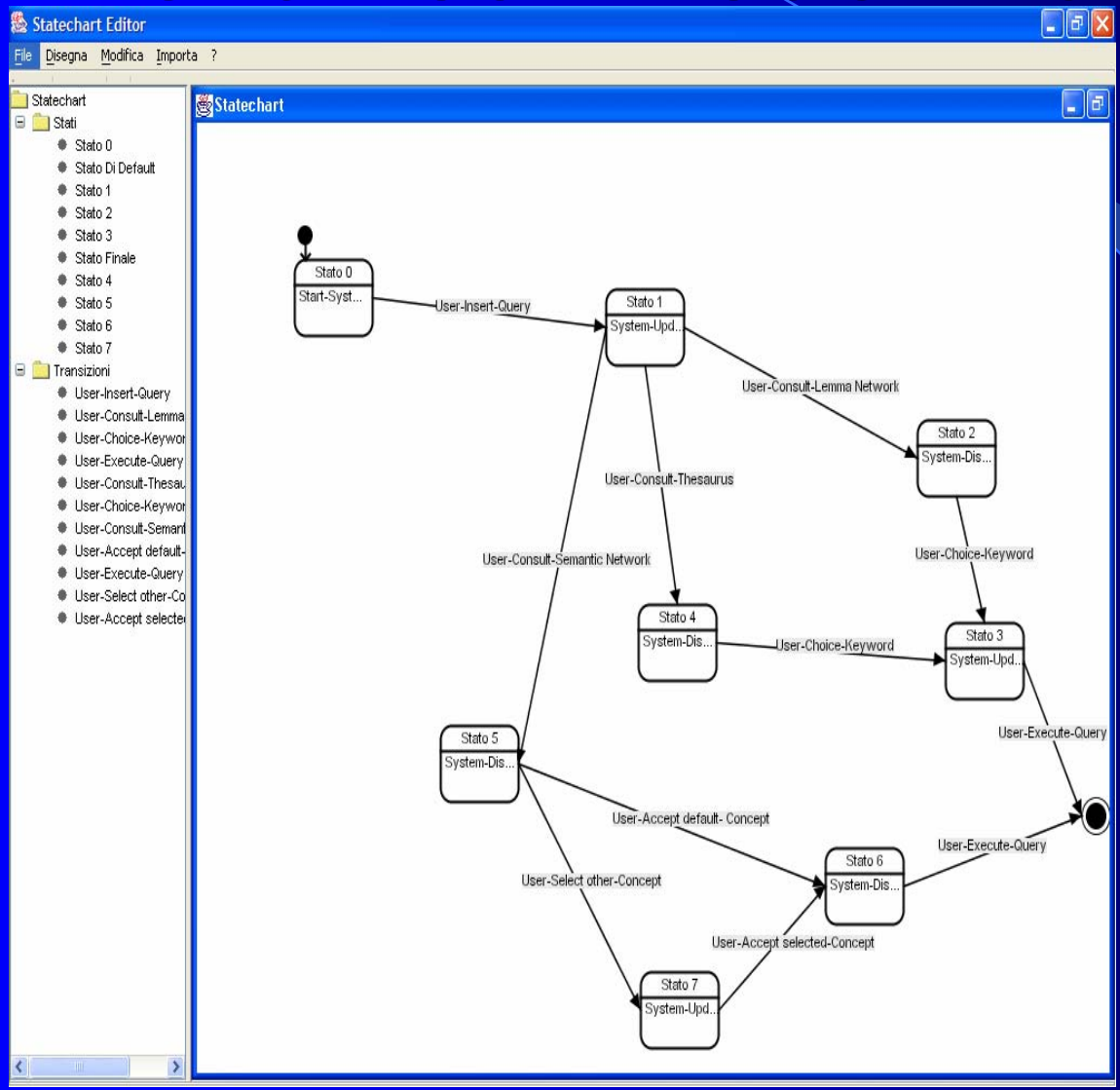
State Diagram View



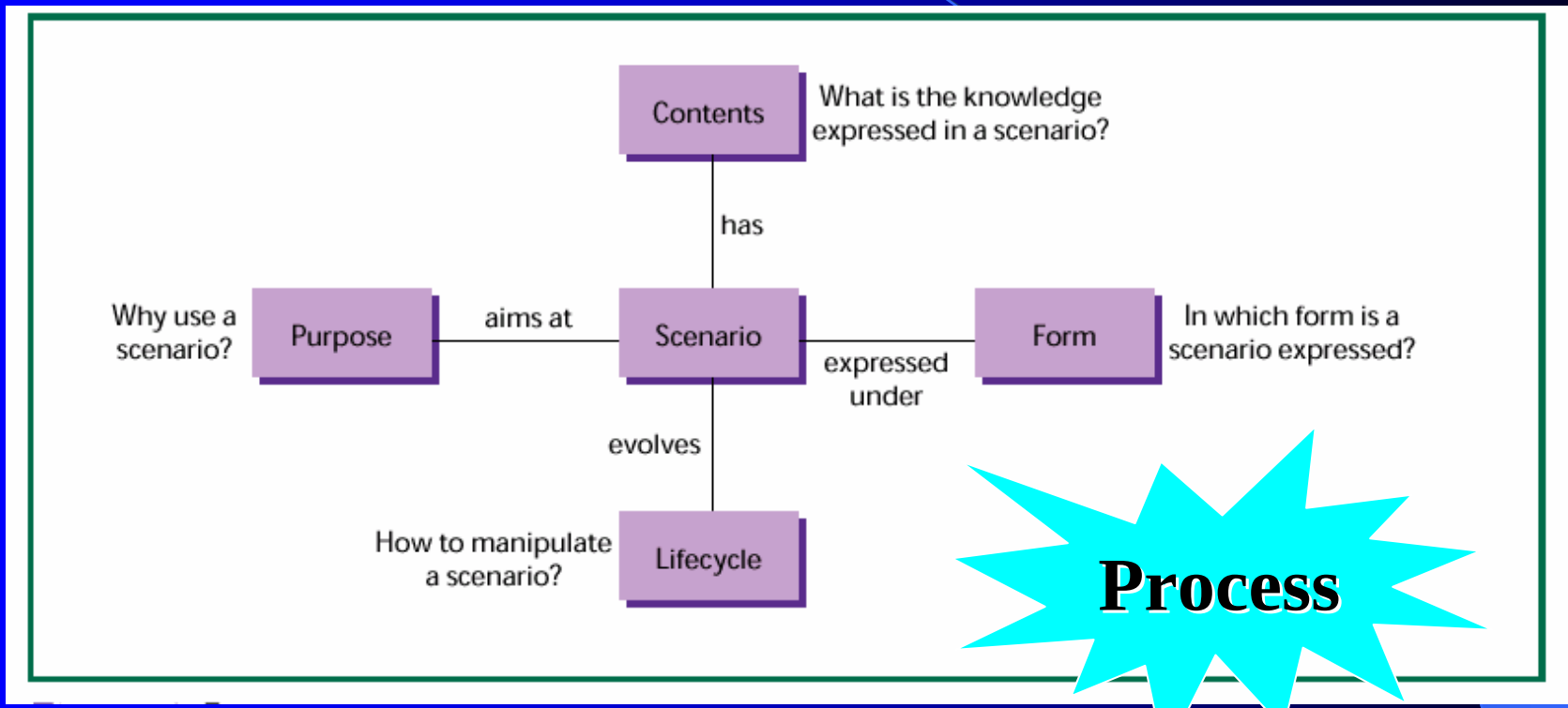
Test Model Manager



Behaviour Model Manager



Classification Framework

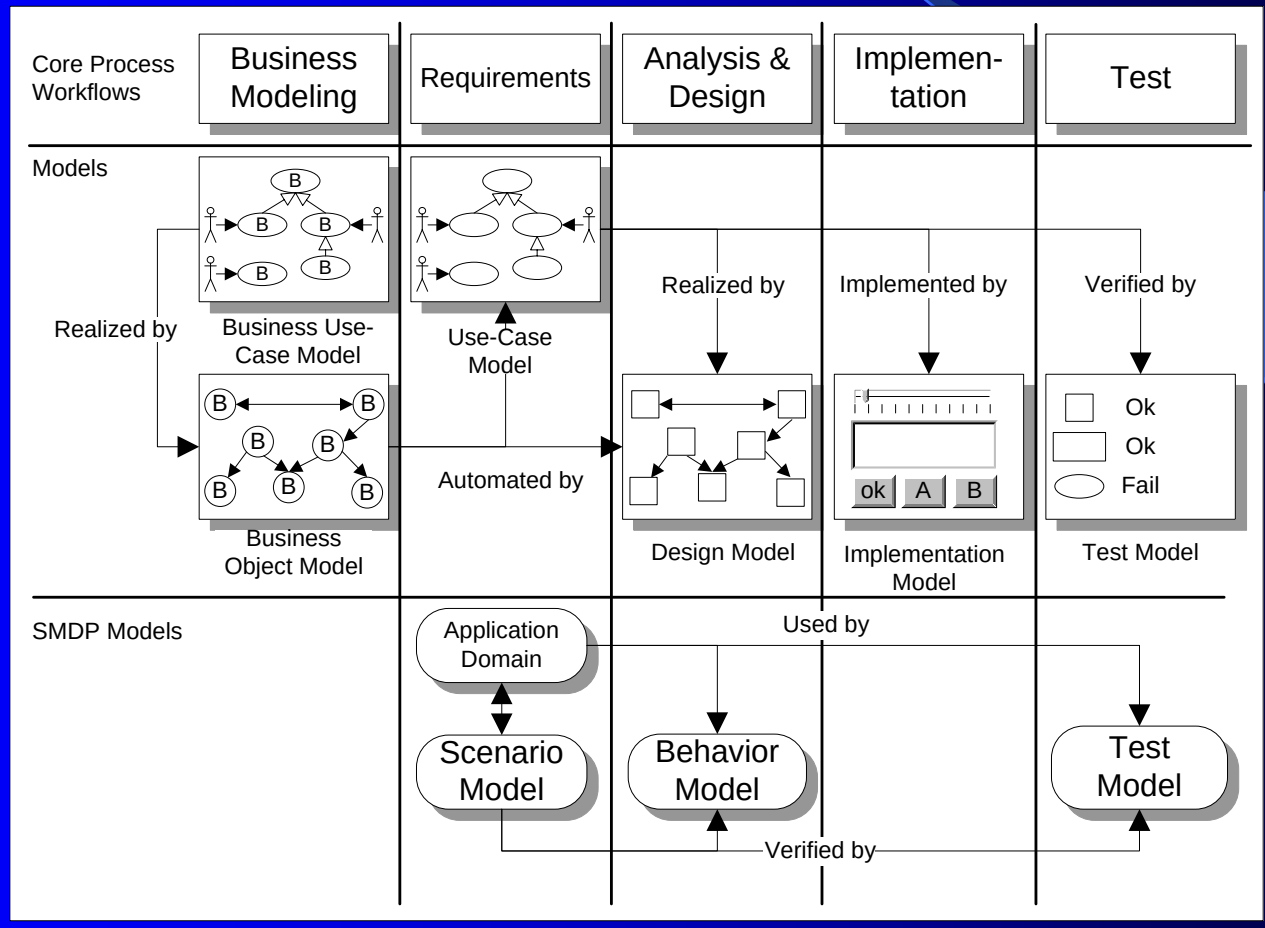


Application of SMDP-SME

Further Research

- to refine and validate SMDP and SME prototype version in order to obtain a real industrial product
- to integrate SME with features of natural language
- to apply our approach in order to document the behaviour of the existent projects before to migrate them using the new technologies
- to support the development of the new project, in particular in the analysis and testing phase
- to document the projects already developed in ordert to create a knowledge repository

END

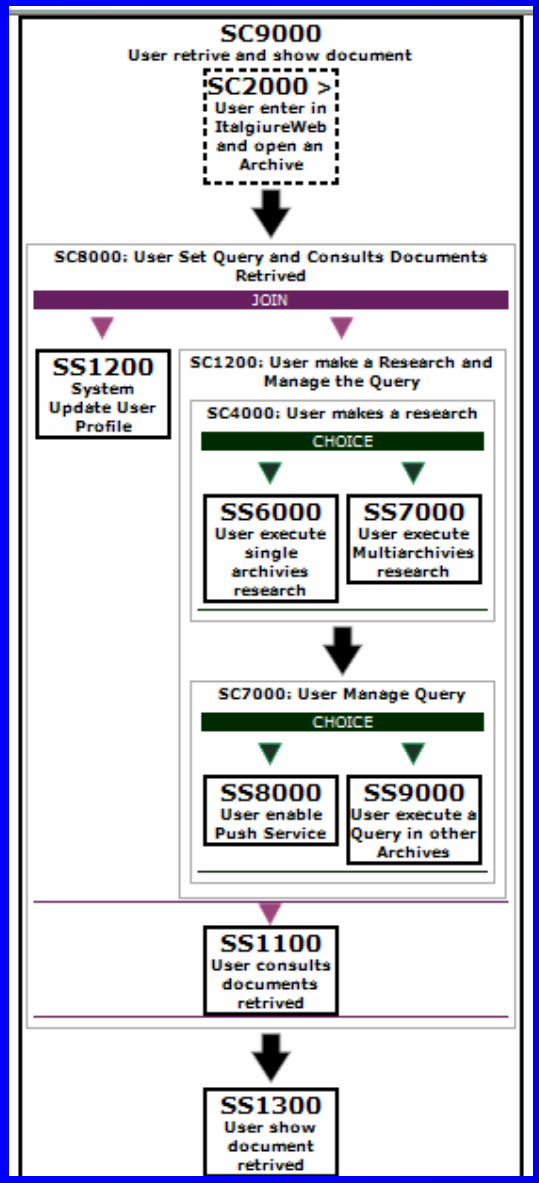

Core Process Workflows
Business Modeling
Requirements
Analysis & Design
Implemen-tation
Test
Models
Realized by
Business Use-Case Model
Use-Case Model
Realized by
Implemented by
Verified by
Business Object Model
Automated by
Design Model
ok
A
B
Ok
Ok
Fail
Implementation Model
Test Model
SMDP Models
Application Domain
Used by
Scenario Model
Behavior Model
Test Model
Verified by

Thesis Contribution

- the definition of a modelling process to support the life cycle and scenario management:
  - the definition of scenario model that includes the composition operators to compose scenarios, without introducing intermediate formalism
  - the definition of a set of transformation rules to derive other form of specification, i.e. test cases
  - the definition of a set of activities
- the development of a software environment to support the methodology
- the application of the process on a wide variety of case studies in order to refine and validate the methodology

Basic Terminology ..continue

- **Scenario**: is an ordered set of interactions that occurs under certain conditions (precondition), to achieve the primary actor's goal.
- **Goal** : is defined as the result expected from the execution of the user system interactions
- **Actors**: they can be a person or another computer system or organization structures that have a role in the scenario
- **Item**: is a relevant physical element or information that must be available in the scenario
- **Trigger**: is an event that causes the execution of the user-system interaction
- **Precondition**: characterises the condition under which the goal may be achieved.
- **Postcondition**: indicates what will be the result after the execution of the sequence of interactions.
- **Step**: corresponds to the interactions between actors (user and system) to achieve the goal of the scenario.


SC9000
User retrive and show document
SC2000 >
User enter in ItalgiureWeb and open an Archive
SC8000: User Set Query and Consults Documents Retrived
JOIN
SS1200
System Update User Profile
SC1200: User make a Research and Manage the Query
SC4000: User makes a research
CHOICE
SS6000
User execute single archivies research
SS7000
User execute Multiarchivies research
SC7000: User Manage Query
CHOICE
SS8000
User enable Push Service
SS9000
User execute a Query in other Archives
SS1100
User consults documents retrived
SS1300
User show document retrived