

User Requirements Notation (URN)

Gregor v. Bochmann, University of Ottawa

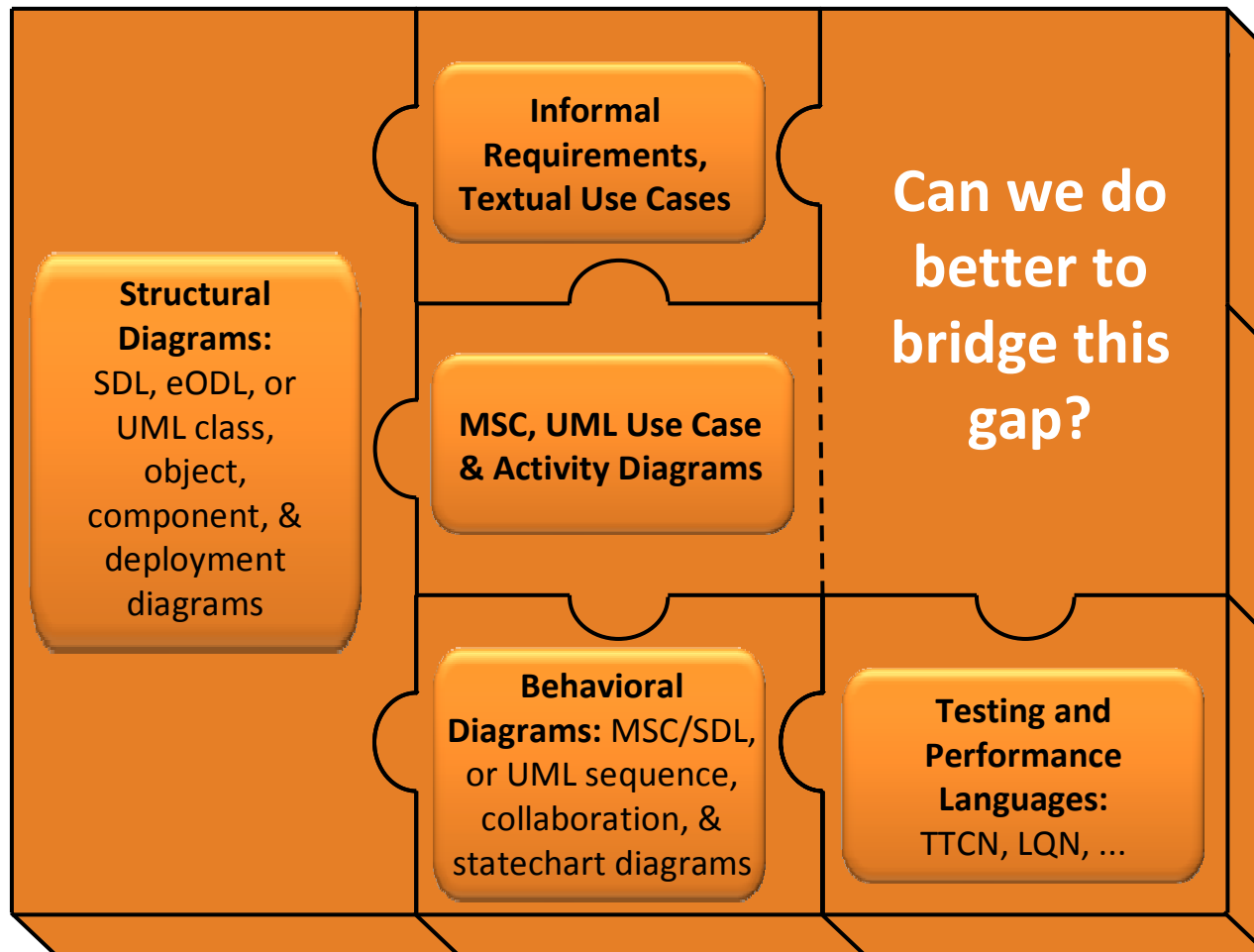
Based on Powerpoint slides by Gunter Mussbacher(2009)
with material from Amyot

Table of Contents

- Introduction to the User Requirements Notation (URN)
 - User Requirements Notation
 - jUCMNav Tool
- Goals and Rationale
- Goal-oriented Requirement Notation (GRL)
 - GRL Basics
 - Evaluations
 - Examples
 - Tools
 - Metamodel
- **All models are false but some models are useful.**¹

[1] George Edward Pelham Box (1919-)

Modeling Puzzle – Common View





Introduction to the User Requirements Notation (URN)

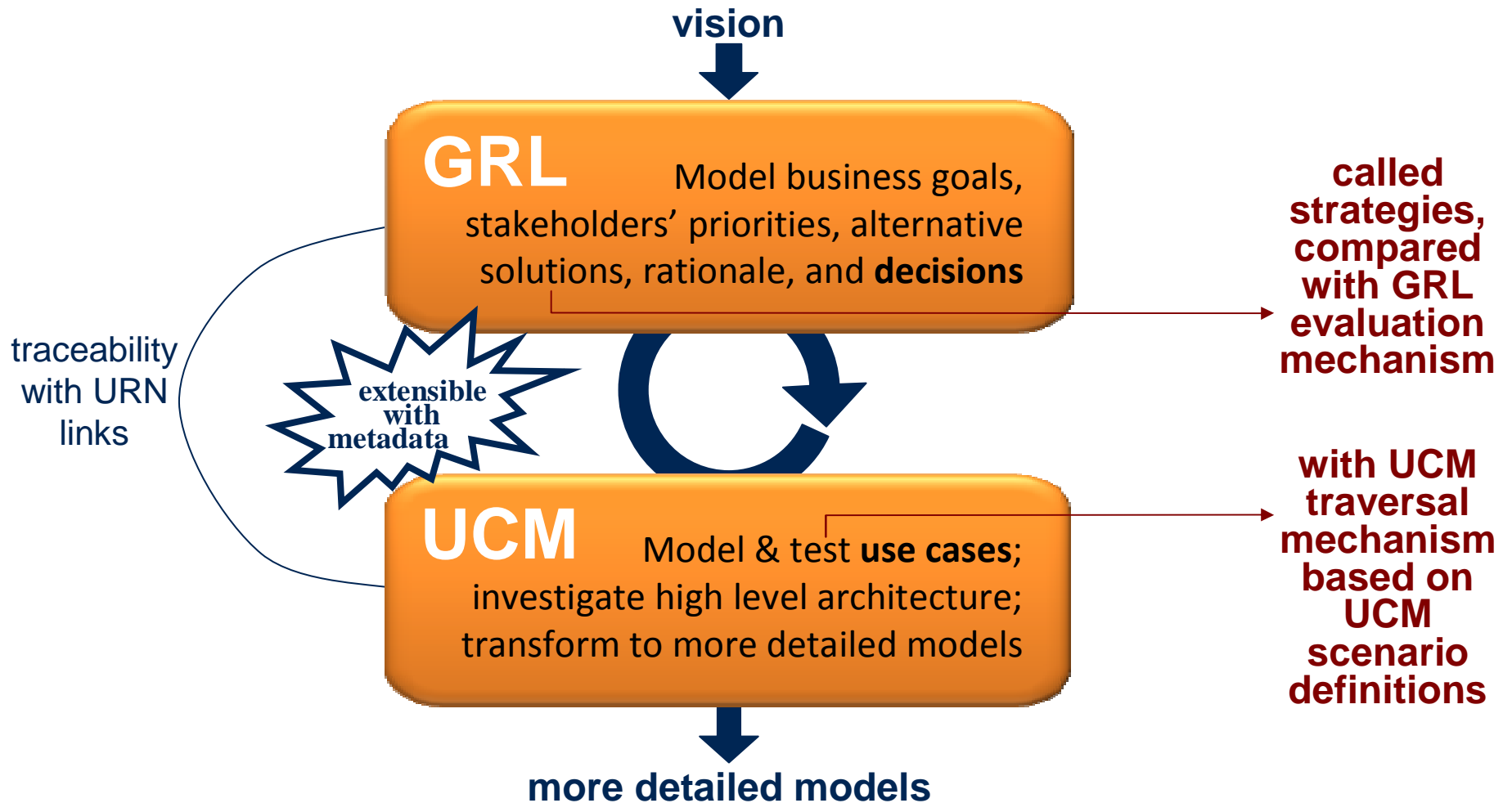
URN Overview (1)

- URN is a semi-formal, lightweight graphical language for modeling and analyzing requirements in the form of goals and scenarios and the links between them
- Combines two existing notations
 - **Goal-oriented Requirement Language (GRL)** (based on i* & NFR Framework)
 - **Use Case Maps (UCMs)**
- Support for the elicitation, analysis, specification, and validation of requirements
- Allows systems, software, and requirements engineers to discover and specify requirements for a proposed system or an evolving system, and analyse such requirements for correctness and completeness

URN Overview (2)

- URN models can be used to specify and analyze various types of reactive systems, business processes, and telecommunications standards
- jUCMNav
 - URN editor & analysis tool
 - Eclipse plug-in
 - Open source project

URN Overview (3)



A GRL / UCM model visually communicates business objectives and constraints / high-level functional requirements to all stakeholders

Combination of Goals and Scenarios

- URN combines two complementary parts:
 - The **Goal-oriented Requirement Language (GRL)**
 - For modelling goals and other intentional concepts (mainly for non-functional requirements, quality attributes, and reasoning about alternatives and tradeoffs)
 - The **Use Case Map (UCM)** notation
 - For modelling scenario concepts (mainly for operational requirements, functional requirements, and performance and architectural reasoning)
- URN offers concepts for the specification of stakeholders, goals, non-functional requirements, rationales, behaviour, scenarios, scenario participants (actors), and high-level architectural structure

URN – Main Objectives

- Focus on early stages of development with **goals** and **scenarios**
- From user requirements to functional and non-functional system requirements
- No messages, components, or component states required
- Reusability
 - of argumentations (goal patterns and analysis)
 - of scenarios (patterns and architectural alternatives)
- Early performance analysis
- Traceability and transformations to other languages
 - Particularly MSC, SDL, TTCN, and UML
 - Also to LQN, LOTOS, and others

ITU-T Z.151: URN – Language Definition

- URN is the **first** and **currently only** standard which explicitly addresses goals (non-functional requirements with GRL) in addition to scenarios (functional requirements with UCMs) in a **graphical** way in one unified language
 - International Telecommunication Union (ITU-T Z.150 series)
 - ITU-T Z.150 (02/03):
User Requirements Notation (URN) - Language requirements and framework
 - ITU-T Z.151 (11/08):
User requirements notation (URN) - Language definition
- Part of the ITU family of languages: SDL, MSC, TTCN-3...
- Definition of URN in Recommendation Z.151 (approved November 2008)
 - Metamodel, abstract/concrete syntaxes, semantics...

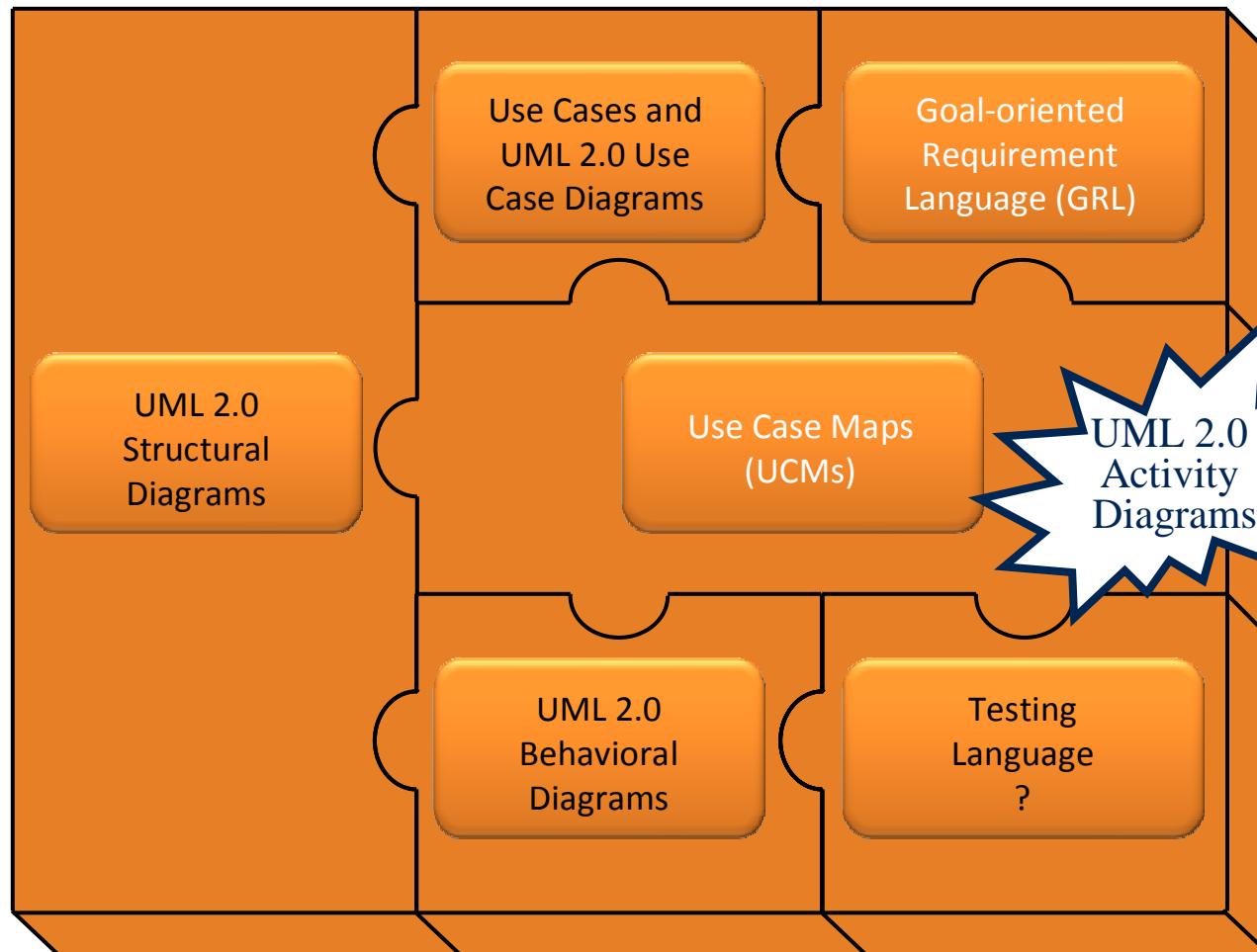
About ITU-T

- International Telecommunication Union
- United Nations organization
 - 191 countries + hundreds of member companies
- Free access to the definition of many standard languages
 - <http://www.itu.int/ITU-T/studygroups/com17/index.html>
- Question 13 (Formal languages and telecommunication software) of Study Group 17 (Security)
 - **Z.15x User Requirements Notation (URN)**
 - Also SDL, MSC, and profiles
- Other international bodies standardize languages
 - ANSI (C, C++), W3C (HTML, XML), IEEE (VHDL), ISO (LOTOS), OMG (UML), OASIS (XACML), etc.

ITU-T Languages

- **SDL**: Specification and Description Language
 - For defining and executing complete reactive systems
- **MSC**: Message Sequence Charts
 - For defining message-oriented scenarios
- **ASN.1**: Abstract Syntax Notation One
 - For defining data types/packets
- **TTCN-3**: Testing and Test Control Notation
 - For defining test cases and test environments
- **URN**: User Requirements Notation
 - Goal-oriented Requirements Language (GRL)
 - Use Case Map notation (UCM)
 - For defining abstract goals, scenarios, and requirements
- **UML 2.0** profiles for some of these languages on their way...

UML Puzzle



GRL captures stakeholders' business goals, alternatives, decisions, and rationales.

Alternatives in GRL are linked with more detailed models described as UCMs.

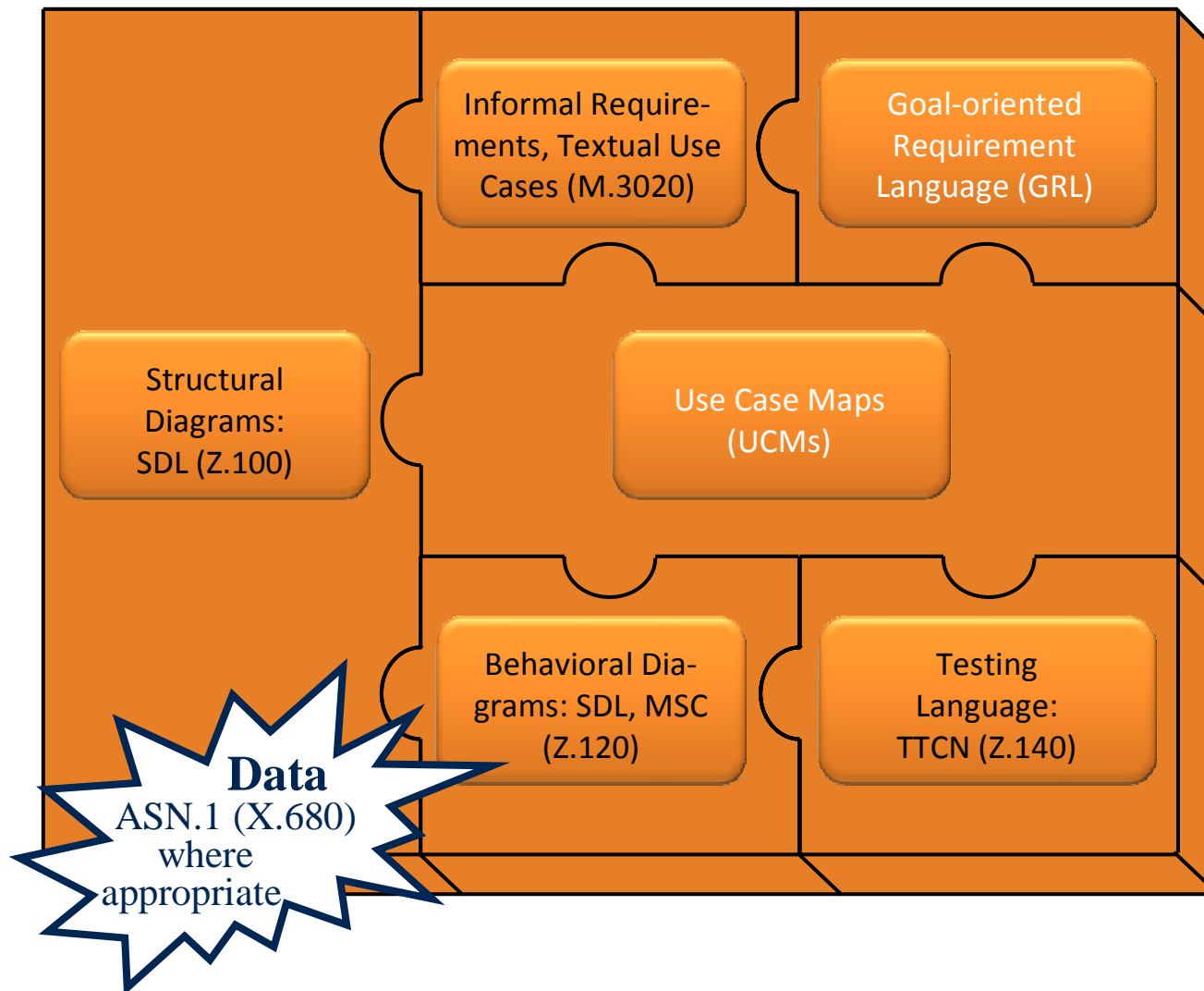
UCMs represent visually use cases in terms of causal responsibilities.

UCMs provide a framework for making high level and detailed design decisions.

UCMs visually associate behavior with structure at the system level.

UCMs enable functional testing and allow the generation of test purposes.

ITU Puzzle



GRL captures stakeholders' business goals, alternatives, decisions, and rationales.

Alternatives in GRL are linked with more detailed models described as UCMs.

UCMs represent visually use cases in terms of causal responsibilities.

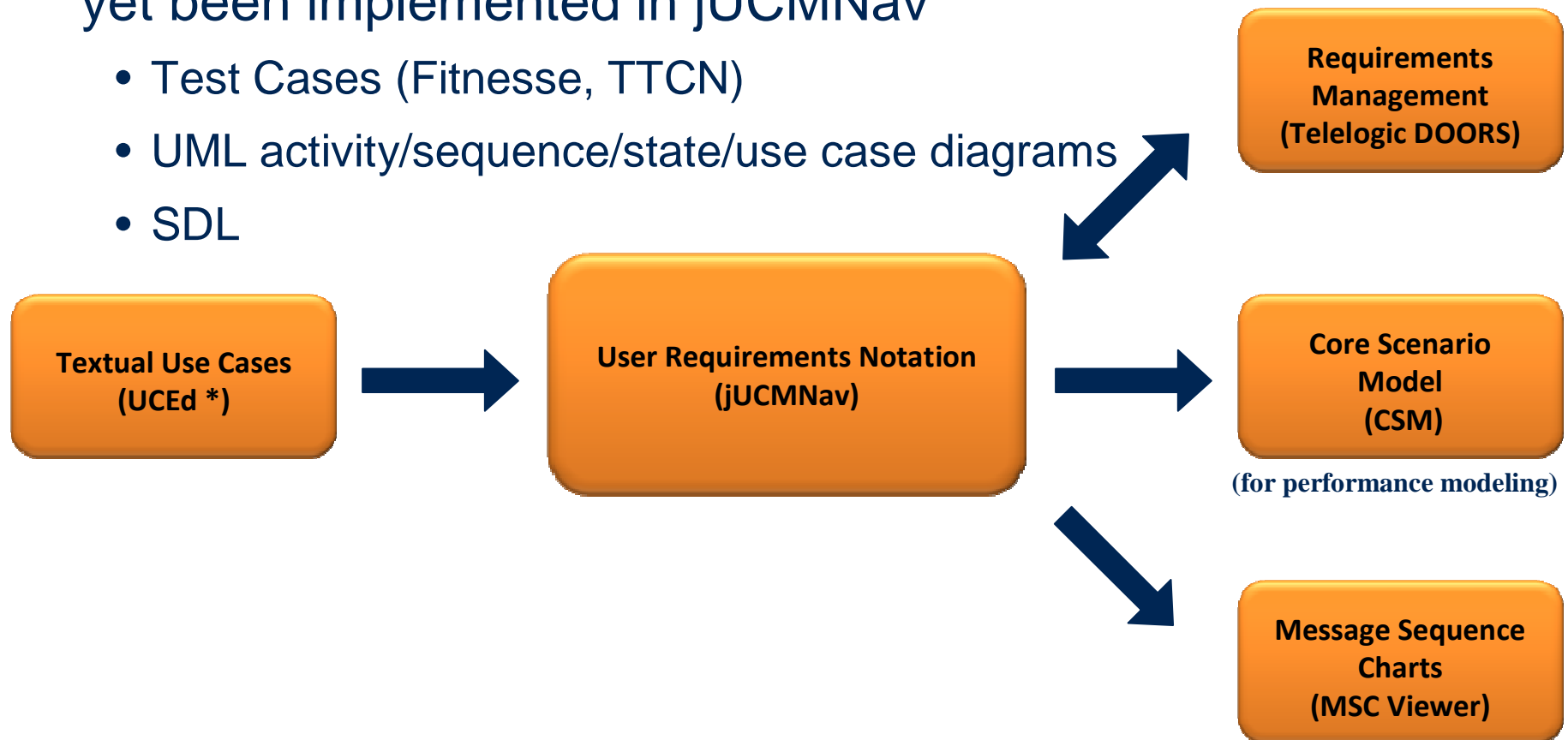
UCMs provide a framework for making high level and detailed design decisions.

UCMs visually associate behavior with structure at the system level.

UCMs enable functional testing and allow the generation of test purposes.

Transformations

- Transformations connect URN models to other models
- Some transformations discussed in publications or implemented in jUCMNav's predecessor UCMNav have not yet been implemented in jUCMNav
 - Test Cases (Fitesse, TTCN)
 - UML activity/sequence/state/use case diagrams
 - SDL



* Stéphane Somé's UCed: <https://sourceforge.net/projects/uced/>

Summary

- The User Requirements Notation (URN) is an ITU-T standard
- Modeling with the Goal-oriented Requirement Language
 - Focuses on answering “**why**” questions
 - Intentions, functional / non-functional requirements, rationales
- Modeling with Use Case Maps
 - Focuses on answering “**what**” questions
 - Scenarios, services, architectures
- While modelling with URN as a whole, goals are **operationalized** into tasks, and tasks are **elaborated** in scenarios (mapped to UCM path elements)
 - Moving towards answering “**how**” questions
 - Can guide the selection of an architecture or scenarios
- Enables the elicitation/specification of systems, standards, & products, analysis from various angles, & transformations

URN Tool: jUCMNav – Juice Up Your Modeling!

- GRL
- Strategies
- Evaluation

- UCMs
- Scenario Definition
- Scenario Execution
- Test Suite

- MSC Generation
- Export to DOORS
- Export to CSM



Pronounced: juicy – em – nav

<http://jucmnav.softwareengineering.ca>

URN Tool: jUCMNav – Overview (1)

- Free (EPL), open-source plug-in for Eclipse
 - Based on Eclipse Modeling Framework (EMF) and the Graphical Editing Framework (GEF)
- Supports most of GRL and UCM notation elements
- Extensible architecture for customized GRL and UCM analysis techniques
- 4 GRL evaluation algorithms, with color highlight
- 1 UCM path traversal mechanism, with export to flat UCMs and MSCs
 - Integrated MSC viewer

URN Tool: jUCMNav – Overview (2)

- Other features
 - Export performance models to CSM
 - Integration with DOORS for requirements management
 - Extensions for Business Process Modeling, Monitoring, and Adaptation
 - Extensions for aspect-oriented modeling with AoURN
 - Verification of user-defined (OCL) rules
 - Report generation (PDF, HTML)
 - Export of diagrams to various bitmap formats

GRL Editor with Strategy Evaluation

The screenshot displays the jUCMNav Eclipse IDE interface for editing GRL models. The main Editor window shows a GRL diagram with nodes and relationships. The diagram is divided into three regions: Service Provider (top), Vendor (middle), and System (bottom). Nodes include "Low Cost (50)", "High Performance (100)", "High Throughput", "Minimum Changes to Infrastructure", "Maximum Hardware Utilisation", "Minimum Message Exchange", "Minimum Switch Load", "Service Nodes Ready For Sale (100)", "Determine Data Location (100)", "Data in Service Control Point", "Data in New Service Node", "Determine Service Location (100)", "Service in Central Switch", "Service in Service Control Point", and "Install Service". Relationships are shown with arrows and labels like "Help", "Hurt", "SomePositive", "SomeNegative", "Break", and "Make".

Key interface components are highlighted with yellow boxes:

- Navigator view:** Located on the left, showing a tree of project elements.
- Outline view:** Located below the Navigator, showing a detailed view of the selected element's structure.
- Editor:** The central workspace for editing the GRL diagram.
- Palette:** Located on the right, providing a library of GRL elements and relationships.
- Scenarios and Strategies view:** Located at the bottom, showing a tree of scenarios and strategies.
- Properties view:** Located at the bottom right, showing the properties of the selected element.

The Properties view for the selected element "Service Nodes Ready For Sale" shows:

- Standard:** Name: Service Nodes Ready For Sale
- Metadata:** (Empty)
- Advanced:** Description: (Empty), Type: Goal

UCM Editor with Scenario Traversal

jUCMNav Execution - URNdocs/IJIS08-WirelessIN-TempFix-Final.jucm - Eclipse

File Edit Navigate Search Project jUCMNav Run Window Help

View all elements

Perspective

Scenarios and Strategie Navigator

UCM Scenarios

- UCMarchAlternatives
 - UCM ServiceAndSDFinSCP_NotOK
 - Included scenarios
 - Start points
 - RootScenario: StartConnection (●)
 - Initializations
 - authorized=false (B)
 - deployment=IN_SCP (E)
 - Preconditions
 - End points
 - RootScenario: Done (102) (|)
 - Postconditions
 - true (x=y)
 - UCM ServiceAndSDFinSCP_OK

Outline

Graphical Outline

Properties Elements

Standard	Property	Value
Metadata	x	245
Metadata	y	154
Advanced	Info	
	description	
	id	271
	name	LogReject
	Parent	ControlFunc...
	Metadata	
	Metadata	[click to edit]
	Miscellaneous	
	hostDemand	

MobileStation

MsgSwitchingCenter

LocationRegister

StartConnection

Reject

Authorization

INI

OUT1

LogReject

Confirmed

CheckLoc

Done

Palette

- Select
- Path Tool (space)
- Comment
- Components
 - Team
 - Object
 - Process
 - Agent
 - Actor
 - Other
- Paths
 - Responsibility (X)
 - Stub (◇)
 - Dynamic Stub (◇)
 - Pointcut Stub (◇)
 - Timer (⌚)
 - Waiting Place (●)
 - Direction Arrow (→)
 - Or Fork (↙)
 - And Fork (⊕)
 - Or Join (↘)
 - And Join (⊖)

GRLmodel RootScenario SvcInMSC ServiceAndSDFinSCP SvcInMSC_DataInSN

Problems

1 error, 0 warnings, 0 others

Description	Resource	Path	Location
Errors (1 item)			
Scenario should have reached end point: ucm.map.impl.EndPointImpl@1fb5a9 (id: 102)	IJIS08-Wire...	/URNdocs	Done

Problems view

Integrated MSC Viewer for Exported UCM Scenarios

The screenshot displays the jUCMNav Eclipse IDE interface. The main window shows a sequence diagram titled "ServiceAndSDFinSCP_OK" with participants: MobileStation, ControlFunction, SDF, SCP, MsgSwitchingCenter, SCF, and LocationDB. The diagram includes messages such as StartConnection, ServiceAndSDFinSCP, GetAuthInfo, ChkAuth, EndPoint, Ok, and CheckLoc. A yellow box labeled "MSC viewer" is overlaid on the diagram.

The Outline view on the left shows a tree structure of the scenario elements, including: ServiceAndSDFinSCP_OK, MobileStation, ControlFunction, SDF, SCP, MsgSwitchingCenter, SCF, LocationDB, LocationRegister, ServiceAndSDFinSCP_OK (expanded), StartConnection, ServiceAndSDFinSCP, GetAuthInfo, ChkAuth, EndPoint, Ok, CheckLoc, and par.

The bottom panel shows a state machine diagram for the MobileStation, MsgSwitchingCenter, and LocationRegister components. The MobileStation component has states StartConnection and Reject. The MsgSwitchingCenter component has states Authorization, LogReject, Confirmed, and Done. The LocationRegister component has states CheckLoc and Done. The diagram shows transitions between these states, including a transition from StartConnection to Authorization, and from Authorization to LogReject, Confirmed, and Done.