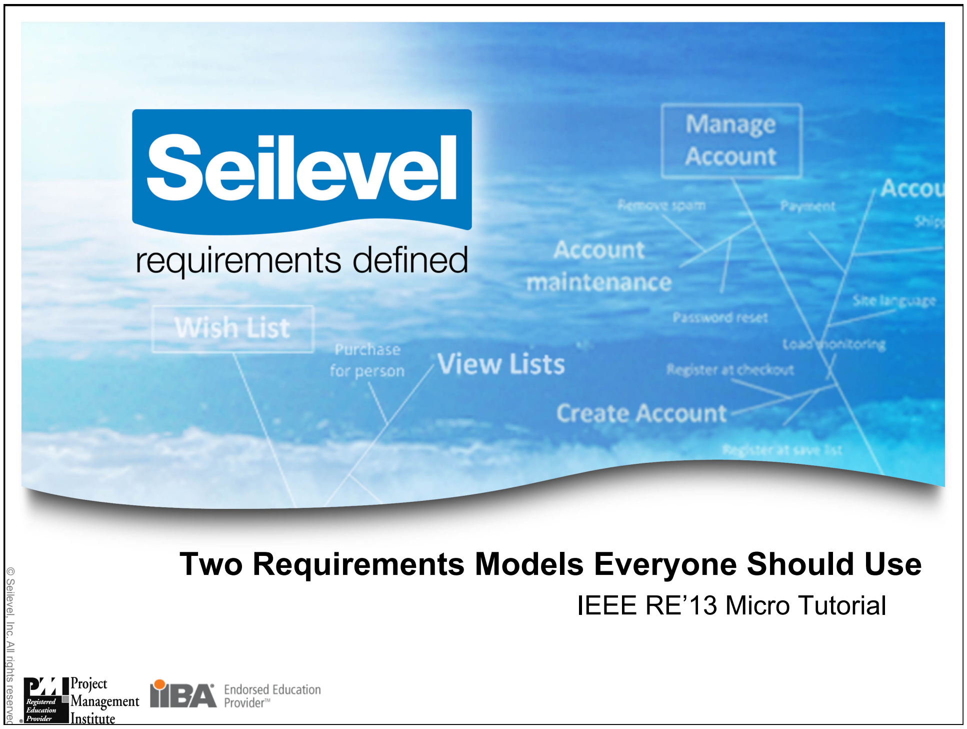
Top Tips You Can Apply Immediately to Projects – Highlights from the RE'13 Tutorials

---

Agenda:
6 tutorial overviews
Save questions for the end!

Implement Visual Models for Software Requirements Immediately (Full Day)	Joy Beatty and/or James Hulgan
Writing Good Requirements (Full day)	Sarah Gregory
Requirements Quality and Productivity Improvement Based on Example Usage	Marcelo Tueiv, Marcelo do Carmo Coelho, Erica Mourão da Silva
Observational and experimental case study research in requirements engineering: Methodology and Examples (Half day)	Roel Wieringa
Model-Based Systems Requirements (Half day - 3 hours).	Bruel Jean-Michel and Joao Araujo
Applying Model Driven Engineering and Domain Specific Languages to Requirements Engineering	Bruce Trask and Angel Roman


Manage Account
Accou
Remove spam
Payment
Ship
Account maintenance
Site language
Password reset
Load monitoring
Wish List
Purchase for person
View Lists
Register at checkout
Create Account
Register at save list

Seilevel

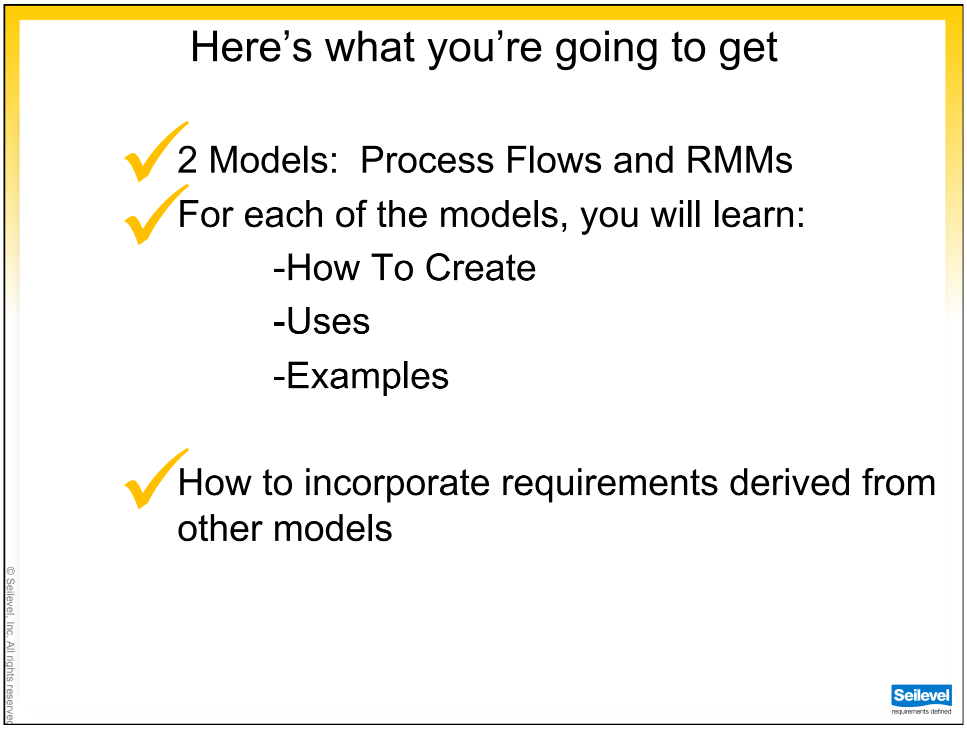
requirements defined

Two Requirements Models Everyone Should Use

IEEE RE'13 Micro Tutorial

PMI Registered Education Provider
Project Management Institute

IIBA Endorsed Education Provider™

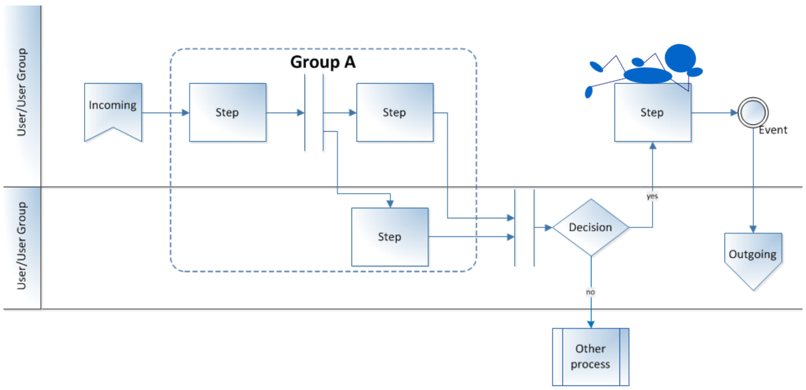


Here's what you're going to get

- ✓ 2 Models: Process Flows and RMMs
- ✓ For each of the models, you will learn:
  - -How To Create
  - -Uses
  - -Examples
- ✓ How to incorporate requirements derived from other models

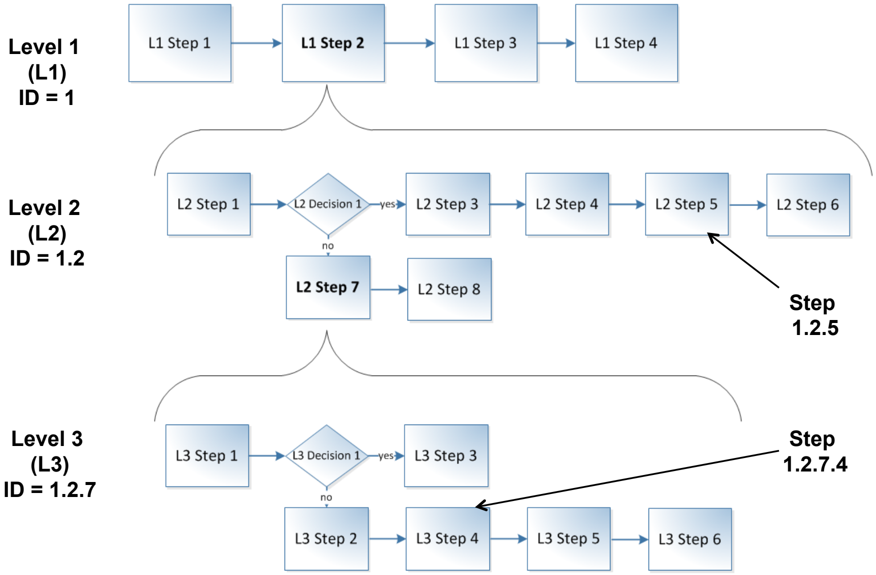
Seilevel
requirements defined

Process Flow



- Show business processes, not system processes (but you can make System Flows too)
- Use in elicitation sessions
- Derive requirements from steps

Process flows have levels: L1, L2, and L3



Level 1
(L1)
ID = 1

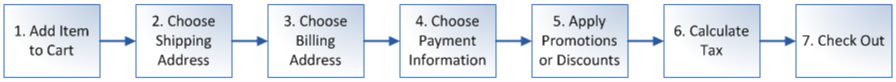
Level 2
(L2)
ID = 1.2

Step
1.2.5

Level 3
(L3)
ID = 1.2.7

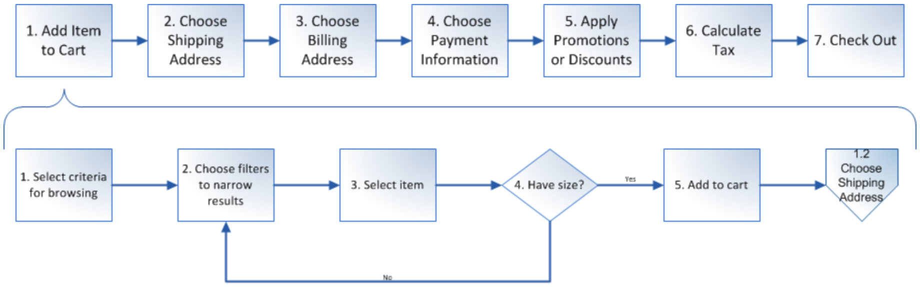
Step
1.2.7.4

Example L1 Process Flow


1. Add Item to Cart
2. Choose Shipping Address
3. Choose Billing Address
4. Choose Payment Information
5. Apply Promotions or Discounts
6. Calculate Tax
7. Check Out

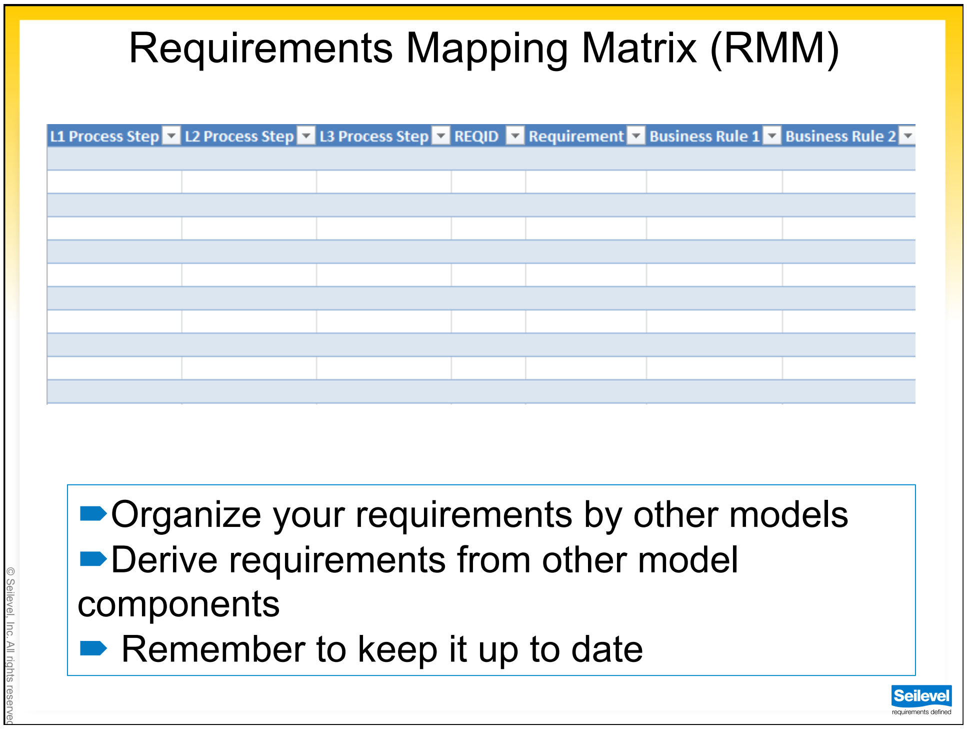
Seilevel
requirements defined

Example L2 Process Flow


1. Add Item to Cart
2. Choose Shipping Address
3. Choose Billing Address
4. Choose Payment Information
5. Apply Promotions or Discounts
6. Calculate Tax
7. Check Out

1. Select criteria for browsing
2. Choose filters to narrow results
3. Select item
4. Have size?
5. Add to cart
1.2 Choose Shipping Address

Seilevel
requirements defined

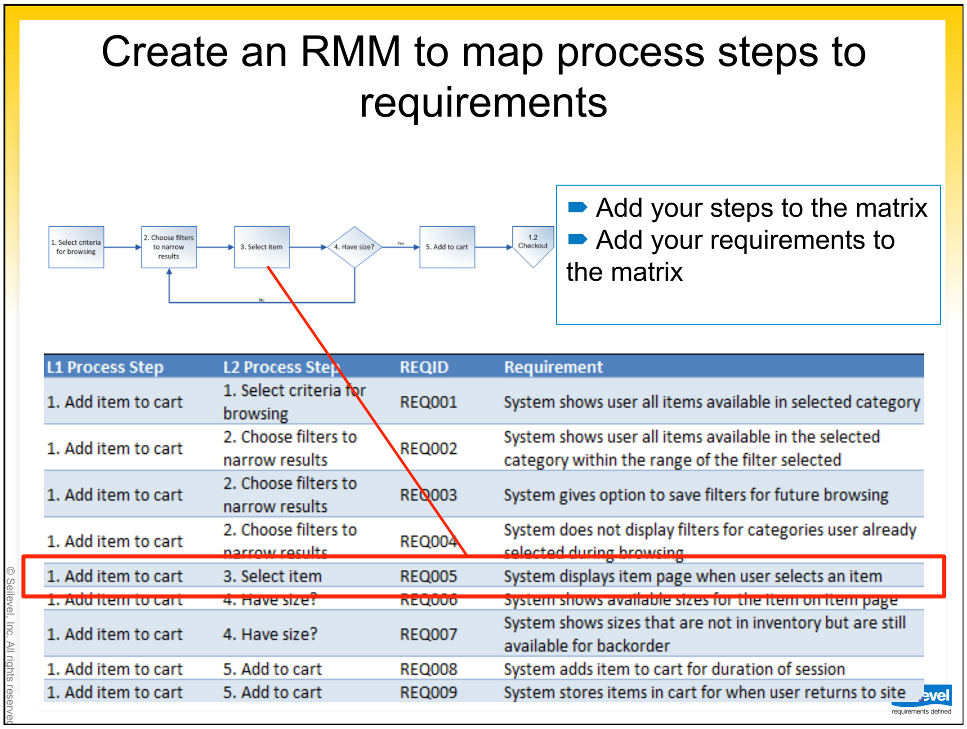


Requirements Mapping Matrix (RMM)

L1 Process Step	L2 Process Step	L3 Process Step	REQID	Requirement	Business Rule 1	Business Rule 2

- Organize your requirements by other models
- Derive requirements from other model components
- Remember to keep it up to date

© Seilevel, Inc. All rights reserved



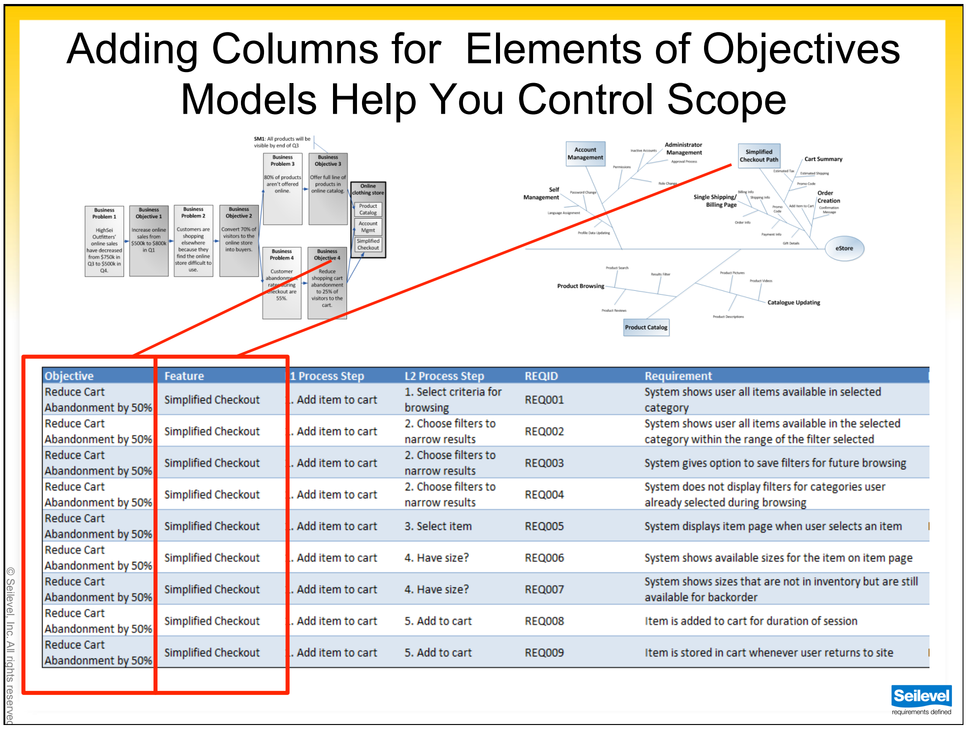
Create an RMM to map process steps to requirements

1. Select criteria for browsing
2. Choose filters to narrow results
3. Select item
4. Have size?
Yes
No
5. Add to cart
1.2 Checkout

- Add your steps to the matrix
- Add your requirements to the matrix

L1 Process Step	L2 Process Step	REQID	Requirement
1. Add item to cart	1. Select criteria for browsing	REQ001	System shows user all items available in selected category
1. Add item to cart	2. Choose filters to narrow results	REQ002	System shows user all items available in the selected category within the range of the filter selected
1. Add item to cart	2. Choose filters to narrow results	REQ003	System gives option to save filters for future browsing
1. Add item to cart	2. Choose filters to narrow results	REQ004	System does not display filters for categories user already selected during browsing
1. Add item to cart	3. Select item	REQ005	System displays item page when user selects an item
1. Add item to cart	4. Have size?	REQ006	System shows available sizes for the item on item page
1. Add item to cart	4. Have size?	REQ007	System shows sizes that are not in inventory but are still available for backorder
1. Add item to cart	5. Add to cart	REQ008	System adds item to cart for duration of session
1. Add item to cart	5. Add to cart	REQ009	System stores items in cart for when user returns to site

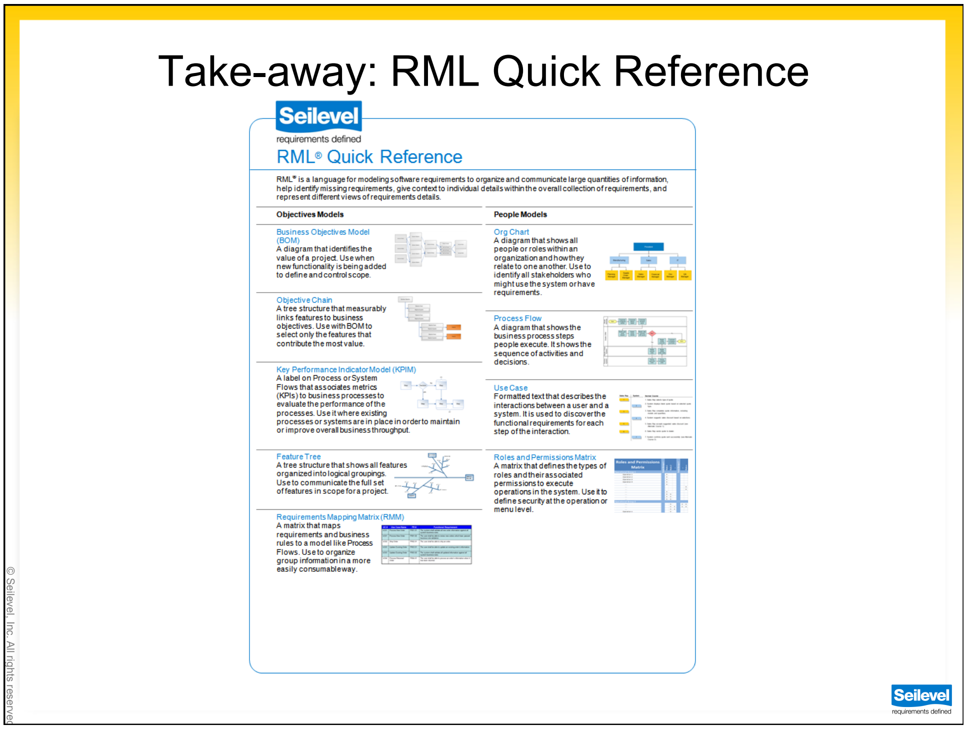
© Seilevel, Inc. All rights reserved



Adding Columns for Elements of Objectives Models Help You Control Scope

Objective	Feature	1 Process Step	L2 Process Step	REQID	Requirement
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	1. Select criteria for browsing	REQ001	System shows user all items available in selected category
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	2. Choose filters to narrow results	REQ002	System shows user all items available in the selected category within the range of the filter selected
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	2. Choose filters to narrow results	REQ003	System gives option to save filters for future browsing
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	2. Choose filters to narrow results	REQ004	System does not display filters for categories user already selected during browsing
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	3. Select item	REQ005	System displays item page when user selects an item
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	4. Have size?	REQ006	System shows available sizes for the item on item page
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	4. Have size?	REQ007	System shows sizes that are not in inventory but are still available for backorder
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	5. Add to cart	REQ008	Item is added to cart for duration of session
Reduce Cart Abandonment by 50%	Simplified Checkout	. Add item to cart	5. Add to cart	REQ009	Item is stored in cart whenever user returns to site

requirements defined



Take-away: RML Quick Reference

requirements defined

RML® Quick Reference

RML® is a language for modeling software requirements to organize and communicate large quantities of information, help identify missing requirements, give context to individual details within the overall collection of requirements, and represent different views of requirements details.

Objectives Models

Business Objectives Model (BOM)
A diagram that identifies the value of a project. Use when new functionality is being added to define and control scope.

Objective Chain
A tree structure that measurably links features to business objectives. Use with BOM to select only the features that contribute the most value.

Key Performance Indicator Model (KPIM)
A label on Process or System Flows that associates metrics (KPIs) to business processes to evaluate the performance of the processes. Use it where existing processes or systems are in place in order to maintain or improve overall business throughput.

Feature Tree
A tree structure that shows all features organized into logical groupings. Use to communicate the full set of features in scope for a project.

Requirements Mapping Matrix (RMM)
A matrix that maps requirements and business rules to a model like Process Flows. Use to organize group information in a more easily consumable way.

People Models

Org Chart
A diagram that shows all people or roles within an organization and how they relate to one another. Use to identify all stakeholders who might use the system or have requirements.

Process Flow
A diagram that shows the business process steps people execute. It shows the sequence of activities and decisions.

Use Case
Formatted text that describes the interactions between a user and a system. It is used to discover the functional requirements for each step of the interaction.

Roles and Permissions Matrix
A matrix that defines the types of roles and their associated permissions to execute operations in the system. Use it to define security at the operation or menu level.

requirements defined

Landing Zone

Contact: sarah.c.gregory@intel.com

The Landing Zone

A Landing Zone is a table that defines a “region” of success for a product or project

The rows of the table contain the subset of requirements that directly define success or failure (*not* all the requirements)

The columns of the table contain a range of performance levels; usually, a Landing Zone covers the range between great success (Outstanding) and failure avoidance (Minimum)

Landing Zones can be used in agile development to help define success of an iteration or Scrum sprint

Landing Zones focus attention on what will create success

Landing Zone Usage

Landing Zones are useful for several things:

- **Gain explicit consensus** at the start of a project on the definition of success
- **Quantify the achievement levels required** as an input to feasibility and risk analysis
- **Drive tradeoff discussions** and decision making throughout the project
- **Monitor and communicate** product attribute status to decision forums and management during development

---

Landing Zone Usage

Landing Zones help clarify decision authority for a team:

Decisions that do not violate any row of the LZ are made by the team as a normal part of their work

- So long as the team meets all LZ rows, that is success

Any decision that would cause any LZ row to be violated requires ratification from a higher authority

- This would include falling below Minimum or a decision to pursue something beyond Outstanding

Landing Zones can be created for platforms, components, service offerings, user experiences, projects, etc.

Exercise 4

Creating a Landing Zone

Instructions:

1. Work individually or in small teams
2. Create a landing zone for you next car purchase, vacation or similar item of your choice; try to include some functions and constraints in addition to rows describing qualities and performance
3. Be ready to share your work with the class when done

Skill taught: Create a simple landing zone for a product or project

---

Landing Zone Length

A good Landing Zone is short enough to be comprehended

- A reasonable guideline is two dozen rows at most, and one dozen is better
- The top-level Landing Zone contains only those topics that must be reviewed regularly by upper management
- Additional "child" Landing Zones can capture details and data on other topics; Tabs of a spreadsheet work well for this

Landing Zone format may provide enough data for precedented, low-risk requirements, but *augment Landing Zone entries with full requirements statements in a separate specification when needed*

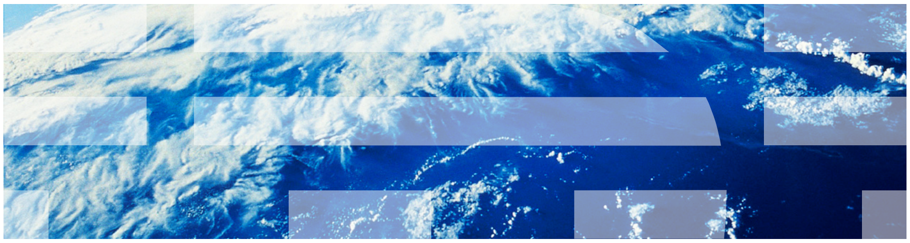
Example:
Car Purchase Landing Zone

Attribute	Minimum	Target	Outstanding
Price	$27000	$20000	$17,500
Mileage (City)	18mpg	25mpg	35mpg
Seating	4 adults	5 adults	6 adults
Interior Noise at 65 mph	74dBA	65dBA	55dBA
Projected 3-year Maintenance Cost	$3000	$2000	$1500

IBM Requirements Center of Competence Brazil
July/2013

Requirements based on examples
IBM Brazil - Requirements Center of Competence

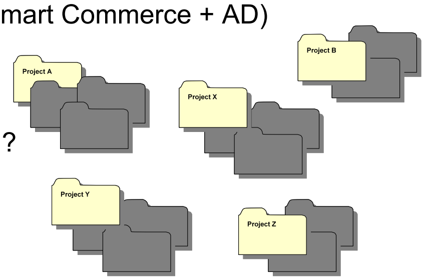
Marcelo Tueiv – mtueiv@br.ibm.com
Erica Mourão da Silva – ericams@br.ibm.com
Marcelo do Carmo Coelho – mcarmo@br.ibm.com



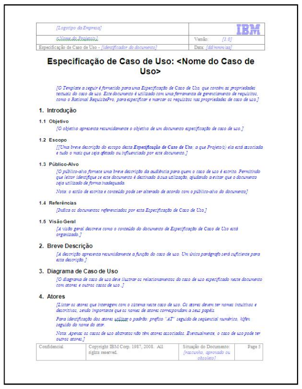
Requirements Team Challenge

How to be more productive / improve quality with:

- Time to Market / Cost Oriented?
- Different Project Approaches ? (e.g.: Smart Commerce + AD)
- Customer Methodologies?
- Industry & Offering Knowledge needed?
- Software Acquisitions?
- Suppliers integrations?
- One of a Kind Projects?
- Writting Communication Issues?
- Hiring New Employees every week?



Project Start, Blank Page



[Logotipo da Empresa]	
<Nome do Projeto>]	Versão: [1.0]
Especificação de Caso de Uso - [identificador do documento]	Data: [dd/mmm/aa]

Especificação de Caso de Uso: <Nome do Caso de Uso>

[O Template a seguir é fornecido para uma Especificação de Caso de Uso, que contém as propriedades textuais do caso de uso. Este documento é utilizado com uma ferramenta de gerenciamento de requisitos, como o Rational RequisitePro, para especificar e marcar os requisitos nas propriedades de caso de uso.]

1. Introdução

1.1 Objetivo

[O objetivo apresenta resumidamente o objetivo de um documento especificação de caso de uso.]

1.2 Escopo

*[[Uma breve descrição do escopo desta **Especificação de Caso de Uso**; a que Projeto(s) ela está associada e tudo o mais que seja afetado ou influenciado por este documento.]*

1.3 Público-Alvo

[O público-alvo fornece uma breve descrição da audiência para quem o caso de uso é escrito. Permitindo que leitor identifique se este documento é destinado à sua utilização, ajudando a evitar que o documento seja utilizado de forma inadequada.

Nota: o estilo de escrita e conteúdo pode ser alterado de acordo com o público-alvo do documento]

1.4 Referências

[Indica os documentos referenciados por esta Especificação de Caso de Uso.]

1.5 Visão Geral

[A visão geral descreve como o conteúdo do documento de Especificação de Caso de Uso está organizado.]

2. Breve Descrição

[A descrição apresenta resumidamente a função do caso de uso. Um único parágrafo será suficiente para esta descrição.]

3. Diagrama de Caso de Uso

[O diagrama de caso de uso deve ilustrar os relacionamentos do caso de uso especificado neste documento com atores e outros casos de uso .]

4. Atores

[Listar os atores que interagem com o sistema neste caso de uso. Os atores devem ter nomes intuitivos e descritivos, sendo importante que os nomes de atores correspondam a seus papéis.

Para identificação dos atores utilizar o padrão: prefixo "AT" seguido de sequencial numérico, hífen seguido do nome do ator.

Nota: Apenas os casos de uso abstratos não têm atores associados. Eventualmente, o caso de uso pode ter outros atores.]

Confidencial	Copyright IBM Corp. 1987, 2008. All rights reserved.	Situação do Documento: [rascunho, aprovado ou obsoleto]	Page 5

- Templates are white boards
  - Try create Performance Rqmt from the scratch
  - Try create Report UC from the scratch

- New Employees without robust guidance
  - Training is not enough

- Teams take time to establish the same level of granularity / quality

- Methods Examples are generic / out of a Context

- Lot of time spent in recurrent Requirements

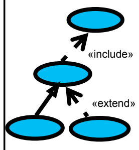
Example Approach

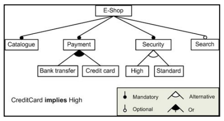
- Example centered
- Approach focused on Reuse of **well estabilished / robust** Examples
- Not focused in cover all scenarios, but to be used as **reference**
- Implements our guidelines
- **Recurrent Requirements** (ex.: Login, Report, Search, Usability etc)
- Use of Requirements Techniques to estabilish **good examples**
  - Questionnaires, Storyboards / Sketchs, UML Diagrams, Quality Requirements Syntaxes
- **Simple to use**

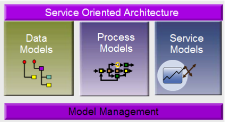
Improve Quality & Productivity

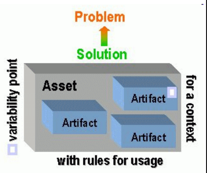
Use of RE & Reuse Approaches

1. Patterns
2. Use Case Fragments
3. Feature Model
4. Industry Frameworks
5. Visual Modeling
6. Requirements Anti-Patterns
7. Asset Based
8. Others (e.g.: Problem Frames)





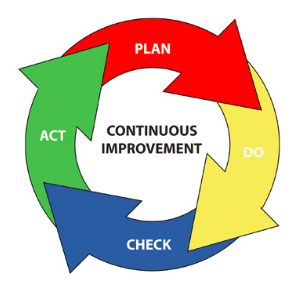




Requirements based on examples - Adoption Plan

Plan

1. Understand Context / Identify Issues
2. Define Goals
3. Define Scope & Approaches
4. Analyze Return of Investment
5. Pilot Project
6. Define Detailed Plan
7. Train Requirements Engineers in Reuse
8. Define Tools / Process
9. Define Rewards Program
10. Define Measurements



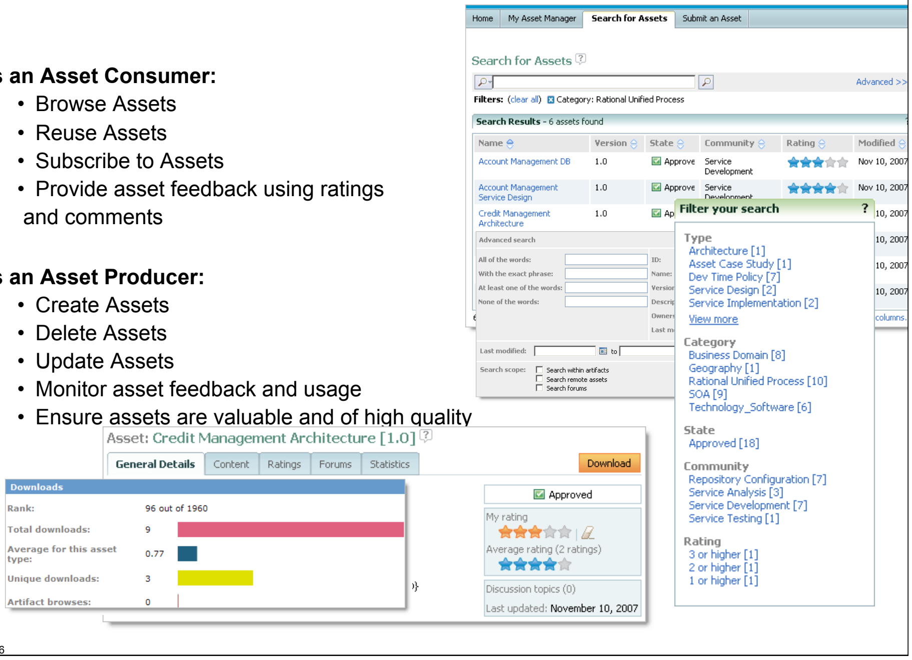
Asset Repository - iRAM

As an Asset Consumer:

- Browse Assets
- Reuse Assets
- Subscribe to Assets
- Provide asset feedback using ratings and comments

As an Asset Producer:

- Create Assets
- Delete Assets
- Update Assets
- Monitor asset feedback and usage
- Ensure assets are valuable and of high quality



Search for Assets

Filters: (clear all) ☒ Category: Rational Unified Process

Search Results - 6 assets found

Name	Version	State	Community	Rating	Modified
Account Management DB	1.0	☑ Approve	Service Development	★★★☆☆	Nov 10, 2007
Account Management Service Design	1.0	☑ Approve	Service Development	★★★★☆	Nov 10, 2007
Credit Management Architecture	1.0	☑ Ap			10, 2007
					10, 2007
					10, 2007
					10, 2007

Advanced search

All of the words:
With the exact phrase:
At least one of the words:
None of the words:
ID:
Name:
Version
Descrip
Owners
Last m
Last modified: to
Search scope: ☐ Search within artifacts ☐ Search remote assets ☐ Search forums

Filter your search ?

Type
Architecture [1]
Asset Case Study [1]
Dev Time Policy [7]
Service Design [2]
Service Implementation [2]
View more

Category
Business Domain [8]
Geography [1]
Rational Unified Process [10]
SOA [9]
Technology_Software [6]

State
Approved [18]

Community
Repository Configuration [7]
Service Analysis [3]
Service Development [7]
Service Testing [1]

Rating
3 or higher [1]
2 or higher [1]
1 or higher [1]

Asset: Credit Management Architecture [1.0]

Downloads

Rank:	96 out of 1960
Total downloads:	9
Average for this asset type:	0.77
Unique downloads:	3
Artifact browses:	0

☑ Approved

My rating ★★★☆☆

Average rating (2 ratings) ★★★★☆

Discussion topics (0)

Last updated: November 10, 2007

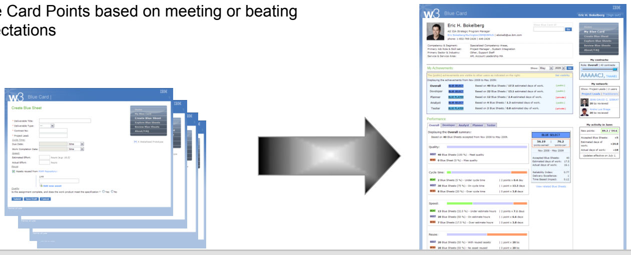
Blue Sheets are used by practitioners to capture individual contribution on project assignments – The results are summarized in their Blue Cards

Blue Sheets

- Document the outcomes of a practitioner's deliverable assignments
- Require self-assessment of four key factors
  - Quality
  - Cycle Time
  - Speed
  - Reuse
- Are validated by the practitioner's project lead
- Earn Blue Card Points based on meeting or beating plan expectations

Blue Cards

- Aggregate completed Blue Sheets from the prior six months
- Quantify a Practitioner's contribution to the business – across multiple projects – in terms of Quality, Cycle Time, Speed and Reuse
- Highlight achievement relative to the broader organization, based on Blue Card Points earned
- Provide an environment where individual accomplishments can be distinguished



+3 Blue Card points for each **Rated Component** reused

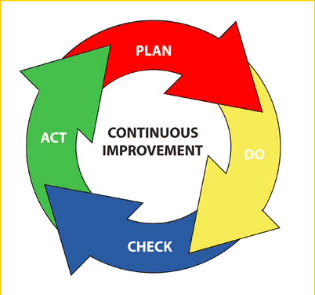
+2 Blue Card points for each **Reviewed Component** reused

+1 Blue Card point for any other asset reused

Requirements based on examples - Adoption Plan

Do – For Each Type of Requirements

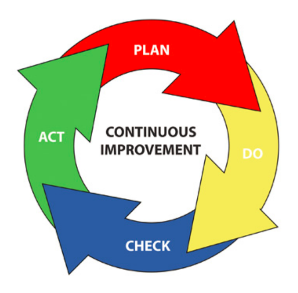
1. Define Examples Meta model
2. Create Requirements Guideline
3. Create Reuse Taxonomy (recurrent rqmnts)
4. Requirements harvest (old projects)
5. Review/Create Examples
6. Package as Asset and Submit for Review
7. Train BAs
8. New Contributions



Requirements based on examples - Adoption Plan

Do – For Each Type of Requirements

1. **Define Examples Meta model**
2. Create Requirements Guideline
3. Create Reuse Taxonomy (recurrent rqmnts)
4. Requirements harvest (old projects)
5. Review/Create Examples
6. Package as Asset and Submit for Review
7. Train BAs
8. New Contributions



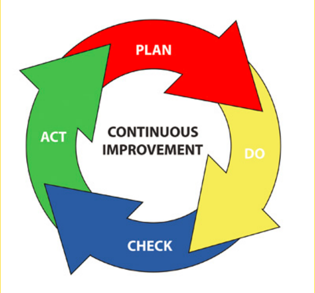
Define Example Meta model

- **Example**
- **Basic details**
- **Applicability**
- **Questionnaire**
- **Discussion**
- **UML Diagrams**
- **Sketch / Storyboard**
- **Anti Patterns**
- **Considerations for development / testing**
- **Related Examples**

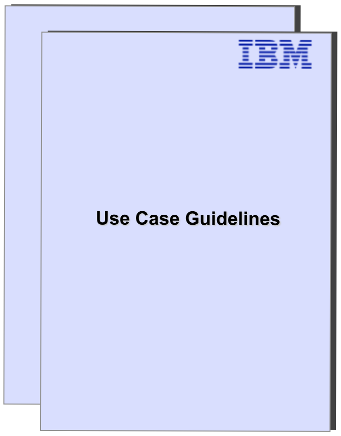
Requirements based on examples - Adoption Plan

Do – For Each Type of Requirements

1. Define Examples Meta model
2. **Create Requirements Guideline**
3. Create Reuse Taxonomy (recurrent rqmnts)
4. Requirements harvest (old projects)
5. Review/Create Examples
6. Package as Asset and Submit for Review
7. Train BAs
8. New Contributions



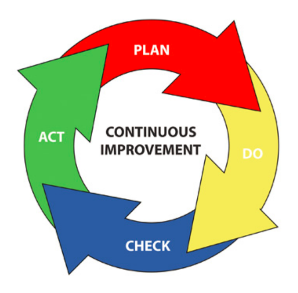
Importance of Guidelines for Reuse



- Definition of
  - Style
  - Granularity
  - Syntax
  - Implement Quality concerns (ambiguity, traceability, testability, completeness etc)
  - Standardization
  - Other concerns for each type of requirement
- Will be used as base for Requirements QA Checklist
- Robust Guidelines provide **Guidance** to Requirements team about Quality Requirements

Requirements based on examples - Adoption Plan

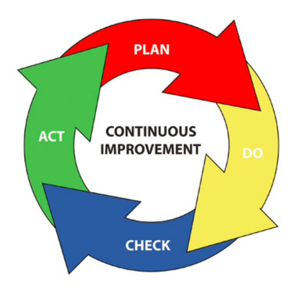
Do – For Each Type of Requirements



1. Define Examples Meta model
2. Create Requirements Guideline
3. **Create Reuse Taxonomy (recurrent rqmnts)**
4. Requirements harvest (old projects)
5. Review/Create Examples
6. Package as Asset and Submit for Review
7. Train BAs
8. New Contributions

Requirements based on examples - Adoption Plan

Do – For Each Type of Requirements

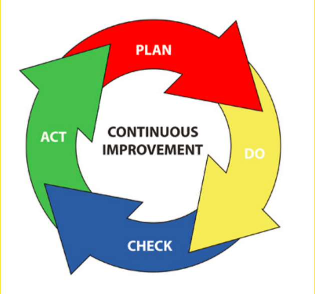


1. Define Examples Meta model
2. Create Requirements Guideline
3. Create Reuse Taxonomy (recurrent rqmnts)
4. **Requirements harvest (old projects)**
5. Review/Create Examples
6. Package as Asset and Submit for Review
7. Train BAs
8. New Contributions

Requirements based on examples - Adoption Plan

Do – For Each Type of Requirements

1. Define Examples Meta model
2. Create Requirements Guideline
3. Create Reuse Taxonomy (recurrent rqrmnts)
4. Requirements harvest (old projects)
5. **Review/Create Examples**
6. Package as Asset and Submit for Review
7. Train BAs
8. New Contributions



Usability based on Examples

Questionnaire:

1. What’s the max number of pages?
2. What ‘s behaviour when no items are found? Is it expected to hide paging options?
3. What shortcuts can be used to navigate in pages?

Discussion:

The navigation bar tells the users the most important information about the list; how many items there are, how many they see now and how to get to the rest. By placing the navigation below the list it is there when users need it most: after scanning all items on the page. Although paging is a very common and accepted way of interacting, the arrival of Ajax technology has introduced new possibilities where paging is no longer needed. All results are simply shown but only loaded as the user scrolls down. See for example the Apple store's software section or Dzone

Considerations for development / testing:

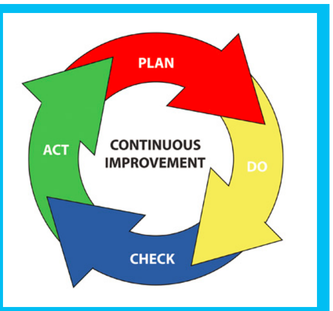
- Deletions of records can bring unexpected behavior of paging. More attention for test these situations: deletion of 1st record, middle or last record.

Source: van Welie, 2005 – Wellie.com

Requirements based on examples - Adoption Plan

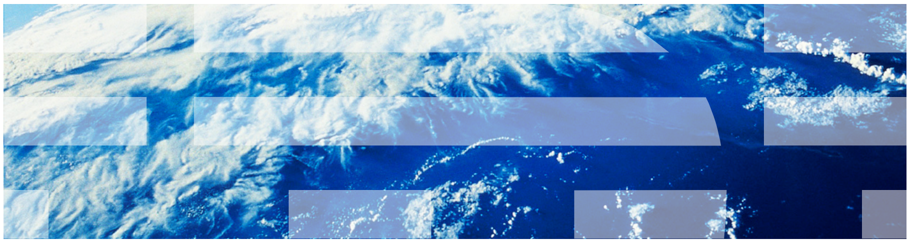
Check – For Each Project

1. Collect Rqrmnts Quality & Examples Metrics
2. Collect Examples Issues/Defect
3. Evaluate Examples Usage
4. Evaluate Examples Contribution
5. Analyze Examples Effectiviness
6. Define Improvement Action Plan



Roel Wieringa

Observational and experimental case study research in RE: Methodology and Examples



What is a case study

- A case is a real-world sociotechnical system
  - Observational case studies (only observe)
  - Technical action research (intervene, using a new technollogy, and observe)

Core message

- You can generalize from a single case study by
  - Observe archictecture of a case;
  - Assess architectural similarity between cases and
  - Estimate whether the mechanism in case with similar architecture, will have similar effects.
- The generalization will be middle-range
  - Not all other cases
  - Realistic abstractions

Architectures and mechanisms

- Case architecture
  - Components
  - Capabilities
  - Mechanisms
- Mechanisms do not occur in isoolation
  - Each mechanism may be understood in isolation
  - No universal law of addition of mechanisms
  - Assess combined effect case by case

Doing a case study

1. Identify the boundary of the case
   - What kind of case? Population
   - What internal structure? (Architecture
2. State your research questions in advance
3. Decide how to collect data in advance
4. Decide how to analyze the data in advance
5. When analyzing:
   - Describe architecture and phenomena
   - Explain (in terms of mechanisms)
   - Generalize (to cases with similar architecture)

Université de Toulouse

FACULDADE DE CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE NOVA DE LISBOA

21st IEEE International
Requirements Engineering Conference
July 15th-19th, 2013. Rio de Janeiro, Brasil.

Model-Based Systems Requirements

Jean-Michel Bruel
João Araújo

JMJ Rio2013

Usability based on Examples

Example:

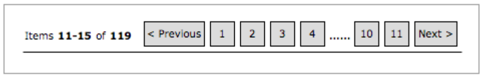
UI0001 – Paging:

The system must present the results grouped in pages with a fixed number of items and allow the users to move from one page of items to another

It will be provided a direct link to a particular page and links to the next/previous page that allow stepping through the pages. Also show the total number of items and use a title to say what kind of items they are.

.

Sketch:



Applicability: Often users need to go through a large list of items. This pattern is applicable when the items are too numerous to fit on one page. The items are typically ordered and the users are likely to find their desired item somewhere near the start, for example in Search Results where this pattern is nearly t always used. Paging is also often used together with a List Builder, for example in an web-based e-mail application. The number of items is typically at 10 to 200 items. The 'Items' can be anything such as e-mail headers, names, photos, phone numbers and so on.

Source: van Welie, 2005 – Wellie.com

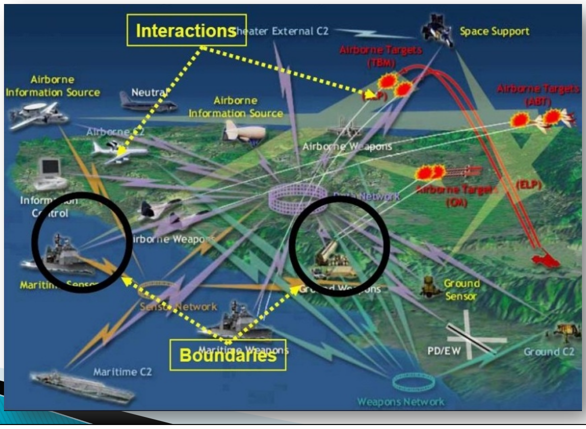
A Complex System

- Set of human and material elements composed of various technologies
  - Computer, Hydraulic, Electronic,…
- Integrated to provide services to its environment corresponding to the system finality
- Interacting between themselves and the environment

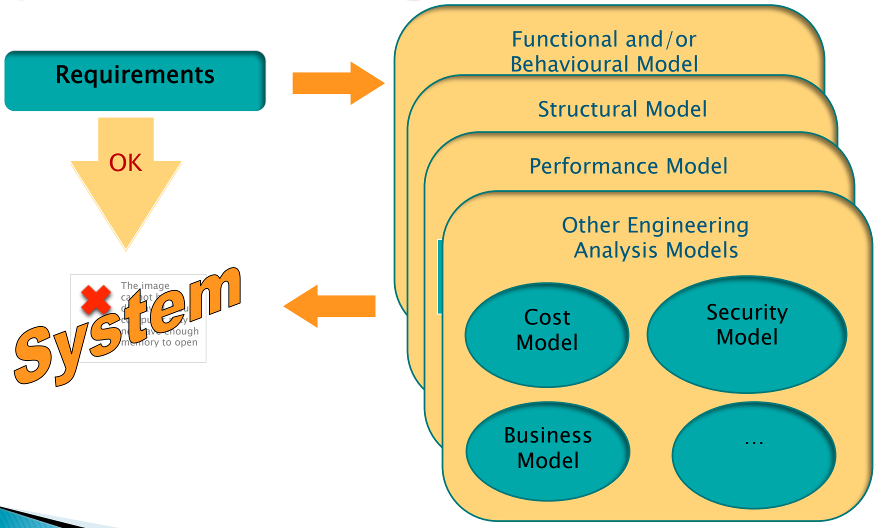
A complex system is very different from a simple software system

Systems of Systems

- A system
  - Should manage interactions between parts
  - Support expected behavior
  - Handle unexpected ones



System Modeling



Requirements

OK

System

Functional and/or Behavioural Model

Structural Model

Performance Model

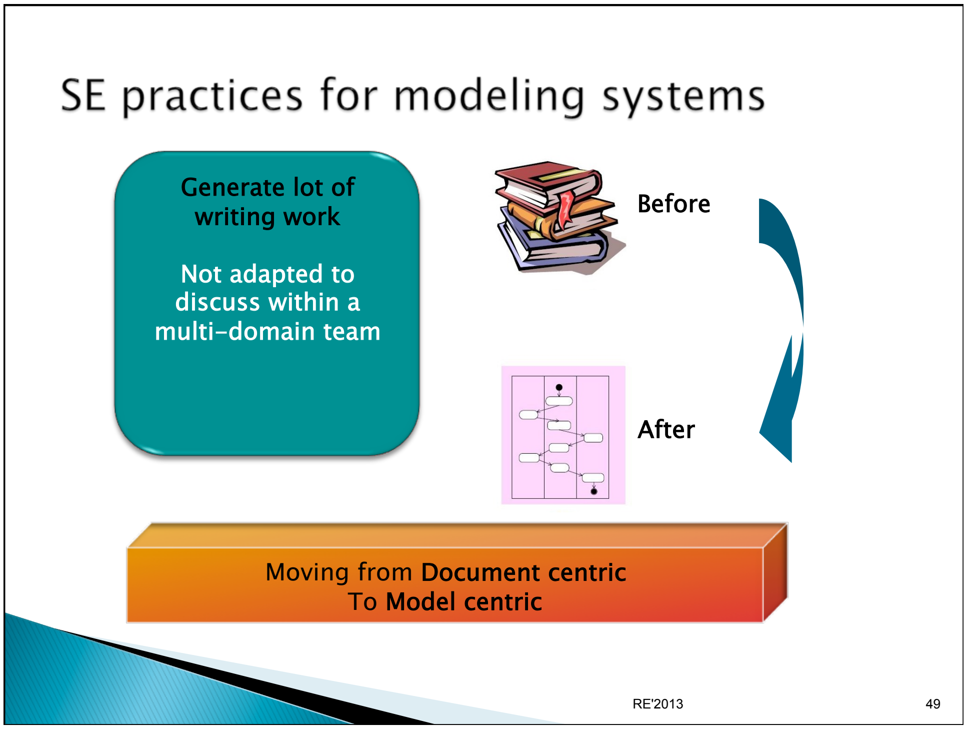
Other Engineering Analysis Models

Cost Model

Security Model

Business Model

…



SE practices for modeling systems

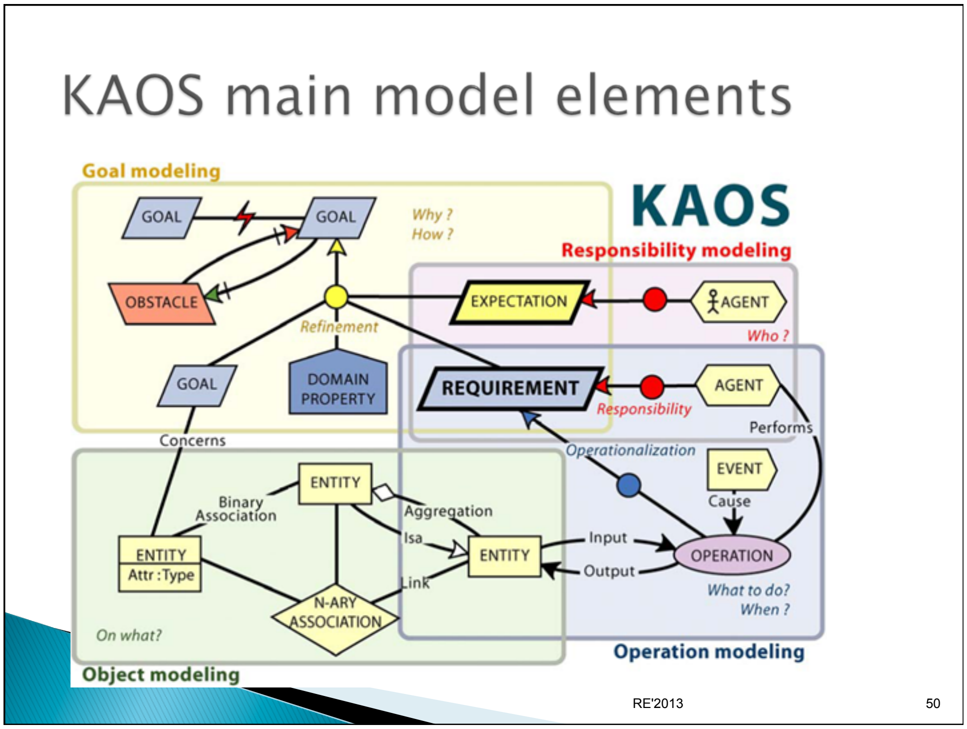
Generate lot of writing work

Not adapted to discuss within a multi-domain team

Before

After

Moving from **Document centric**
To **Model centric**



KAOS main model elements

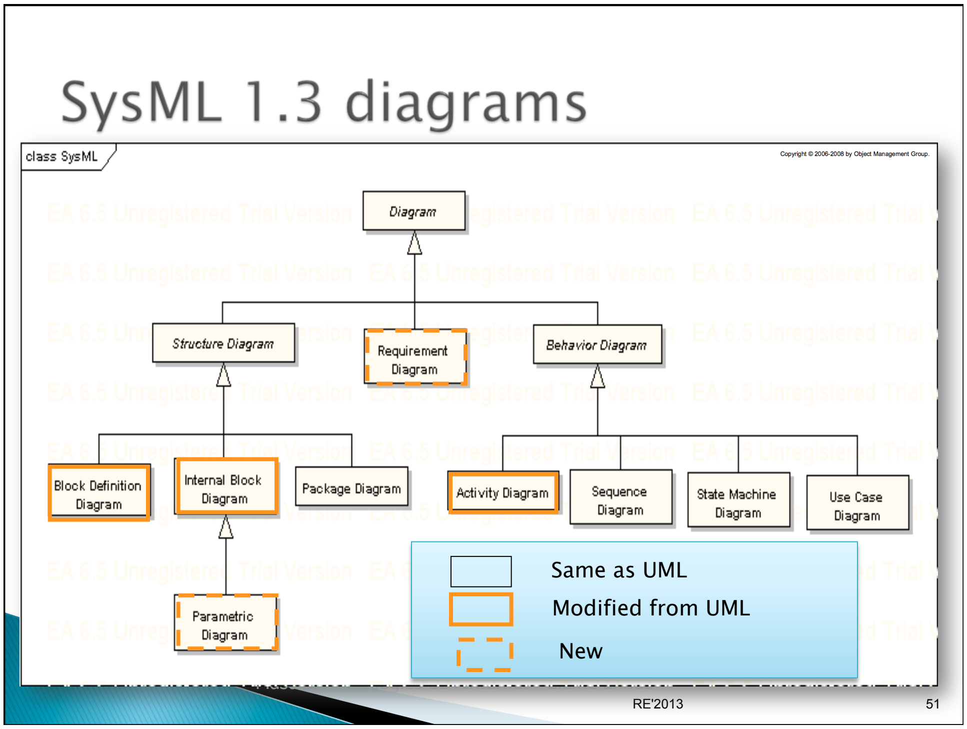
Goal modeling

KAOS

Responsibility modeling

Operation modeling

Object modeling



SysML 1.3 diagrams

class SysML

Diagram

Structure Diagram

Requirement Diagram

Behavior Diagram

Block Definition Diagram

Internal Block Diagram

Package Diagram

Parametric Diagram

Activity Diagram

Sequence Diagram

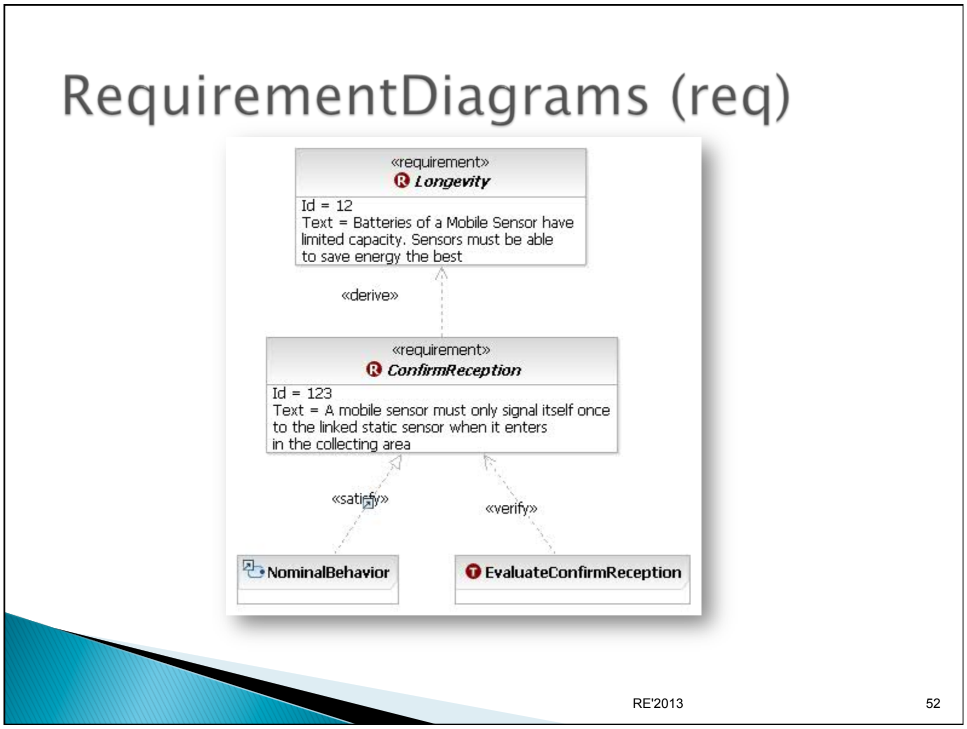
State Machine Diagram

Use Case Diagram

Same as UML

Modified from UML

New



RequirementDiagrams (req)

«requirement»
Longevity
Id = 12
Text = Batteries of a Mobile Sensor have limited capacity. Sensors must be able to save energy the best

«derive»

«requirement»
ConfirmReception
Id = 123
Text = A mobile sensor must only signal itself once to the linked static sensor when it enters in the collecting area

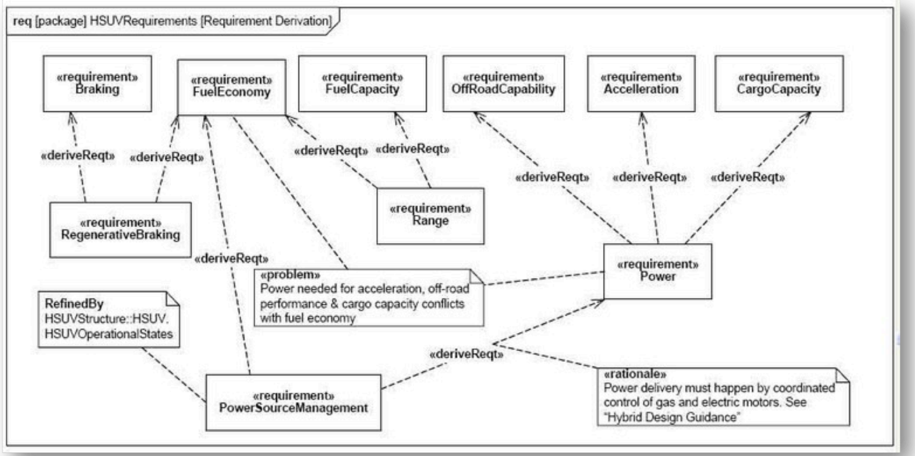
«satisfy»

«verify»

NominalBehavior

EvaluateConfirmReception

Requirement Derivation



req [package] HSUVRequirements [Requirement Derivation]

«requirement» Braking

«requirement» FuelEconomy

«requirement» FuelCapacity

«requirement» OffRoadCapability

«requirement» Accelleration

«requirement» CargoCapacity

«deriveReqt»

«requirement» RegenerativeBraking

«requirement» Range

«requirement» Power

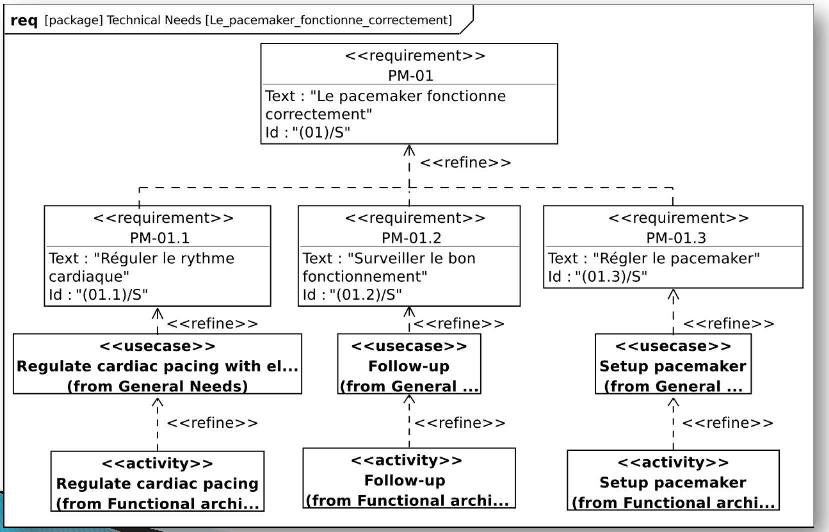
«problem»
Power needed for acceleration, off-road performance & cargo capacity conflicts with fuel economy

RefinedBy
HSUVStructure::HSUV.
HSUVOperationalStates

«requirement» PowerSourceManagement

«rationale»
Power delivery must happen by coordinated control of gas and electric motors. See "Hybrid Design Guidance"

Reducing ambiguity



req [package] Technical Needs [Le_pacemaker_fonctionne_correctement]

<<requirement>>
PM-01
Text : "Le pacemaker fonctionne correctement"
Id : "(01)/S"

<<refine>>

<<requirement>>
PM-01.1
Text : "Réguler le rythme cardiaque"
Id : "(01.1)/S"

<<requirement>>
PM-01.2
Text : "Surveiller le bon fonctionnement"
Id : "(01.2)/S"

<<requirement>>
PM-01.3
Text : "Régler le pacemaker"
Id : "(01.3)/S"

<<usecase>>
Regulate cardiac pacing with el...
(from General Needs)

<<usecase>>
Follow-up
(from General ...

<<usecase>>
Setup pacemaker
(from General ...

<<activity>>
Regulate cardiac pacing
(from Functional archi...

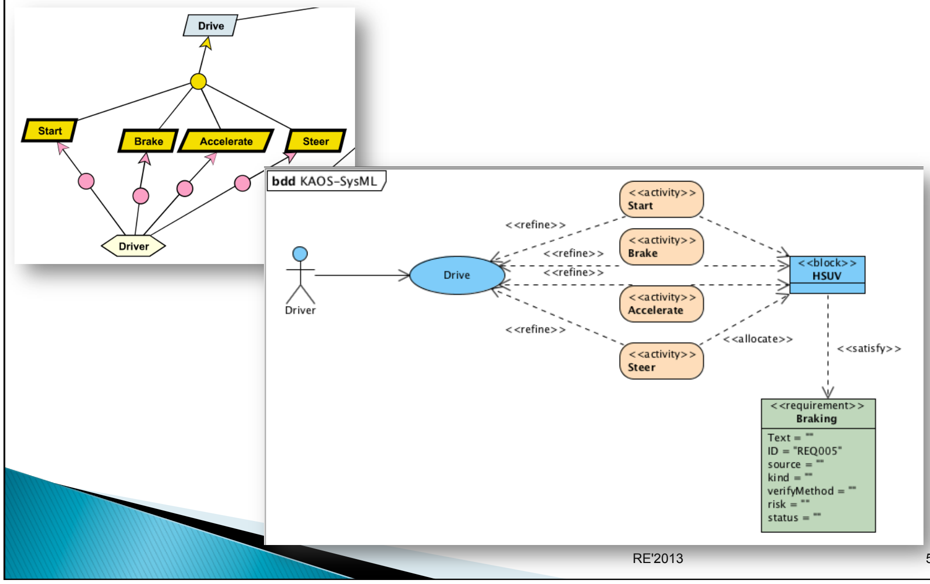
<<activity>>
Follow-up
(from Functional archi...

<<activity>>
Setup pacemaker
(from Functional archi...

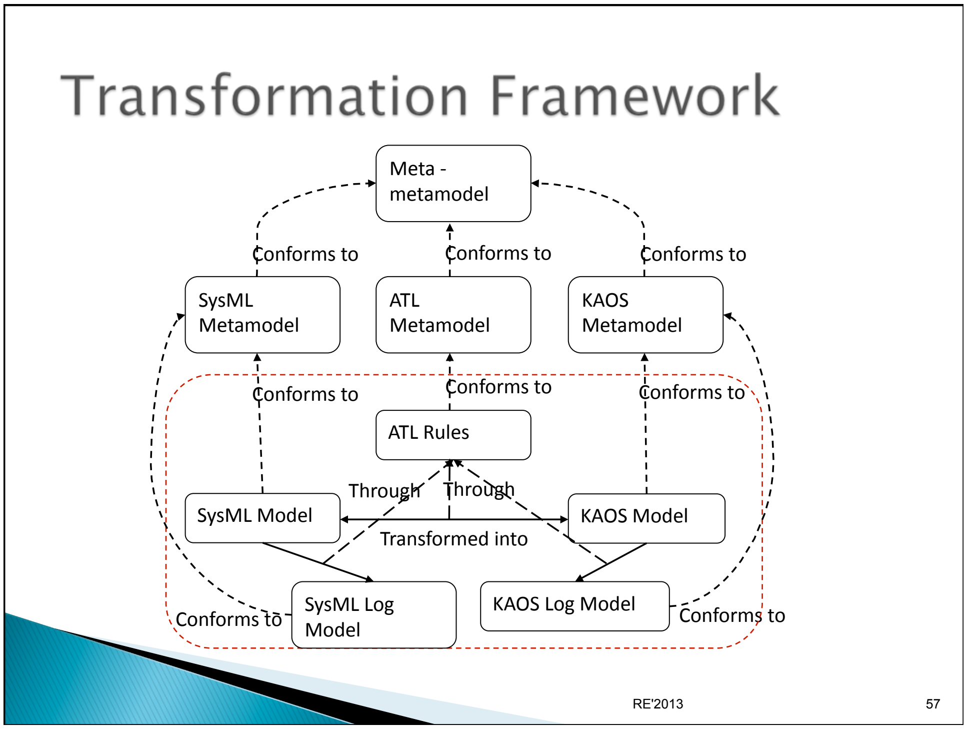
Mapping KAOS models into SysML models

- Mapping Modeling concepts
  - Goal → <<requirement>>
  - Requirement → <<requirement>> (system)
  - Expectation → <<requirement>> (user)
  - Resolutions → <<requirement>> (system or user)
  - Entity → Block
  - Operation → activity or Block operation
  - Environment Agents → Actors
  - System Agents → Blocks/components
  - …
- Relationships
  - Decomposition
    - Or → multiple <<refine>>
    - And → composition
  - Concerns → <<satisfy>>
  - …
- No direct mapping
  - Obstacles
  - Conflicts

Mapping KAOS/SysML: example



Text = ""
ID = "REQ005"
source = ""
kind = ""
verifyMethod = ""
risk = ""
status = ""



Transformation Framework

Meta - metamodel

Conforms to

Conforms to

Conforms to

SysML Metamodel

ATL Metamodel

KAOS Metamodel

Conforms to

Conforms to

Conforms to

ATL Rules

Through

Through

SysML Model

KAOS Model

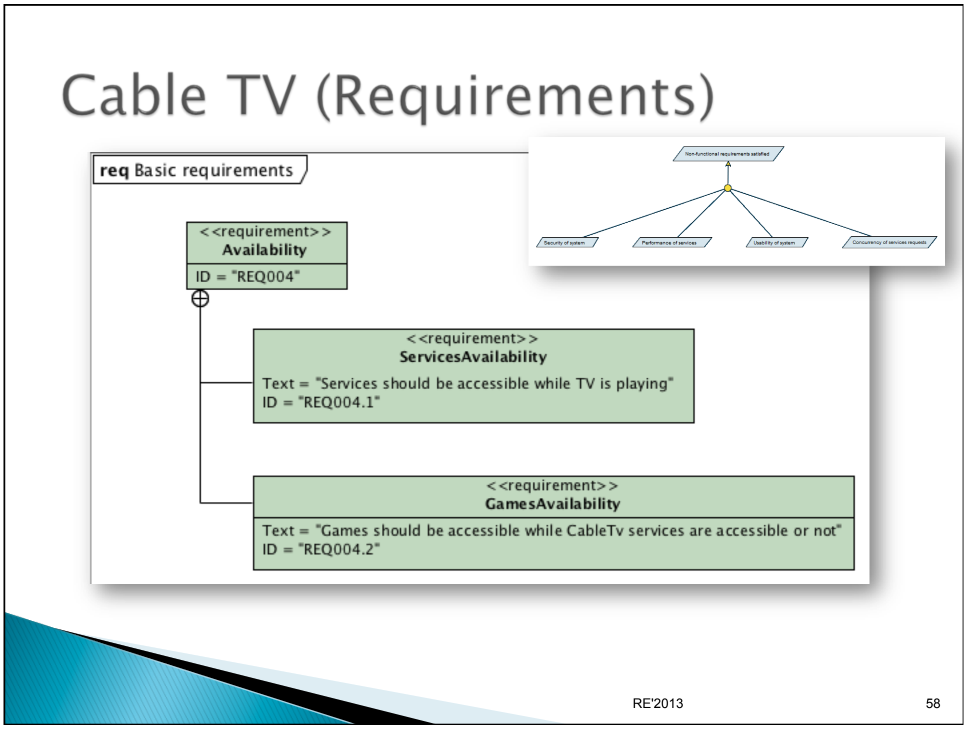
Transformed into

Conforms to

SysML Log Model

KAOS Log Model

Conforms to



Cable TV (Requirements)

req Basic requirements

<<requirement>>
Availability
ID = "REQ004"

<<requirement>>
ServicesAvailability
Text = "Services should be accessible while TV is playing"
ID = "REQ004.1"

<<requirement>>
GamesAvailability
Text = "Games should be accessible while CableTv services are accessible or not"
ID = "REQ004.2"

Non-functional requirements satisfied

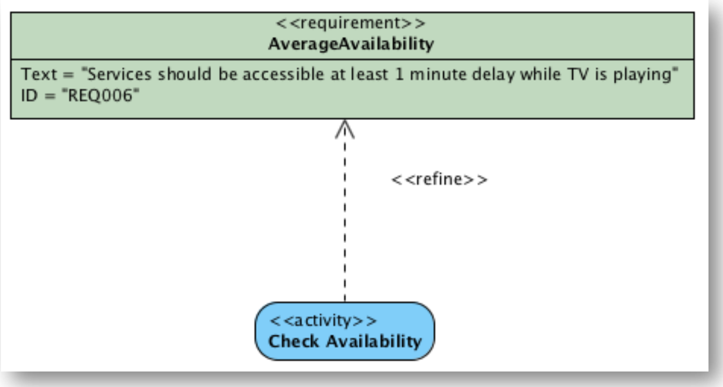
Security of system

Performance of services

Usability of system

Concurrency of services requests

Cable TV (Traceability links)



<<requirement>>
AverageAvailability
Text = "Services should be accessible at least 1 minute delay while TV is playing"
ID = "REQ006"

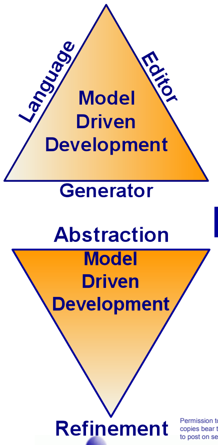
<<refine>>

<<activity>>
Check Availability

Conclusions

- KAOS
  - Goal-oriented modeling language
  - Special role at Requirements elicitation
- SysML is:
  - a specific language for complex systems
  - strongly UML-Based
  - focusing on analysis
  - SysML is not:
    - a method
    - just a UML profile
    - sufficient in itself
- Synergy between KAOS and SysML!
  - Model transformations

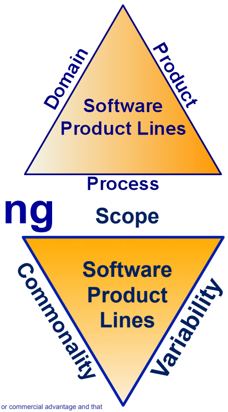
Applying Model Driven Engineering and Domain Specific Languages to Requirements Engineering



Bruce Trask

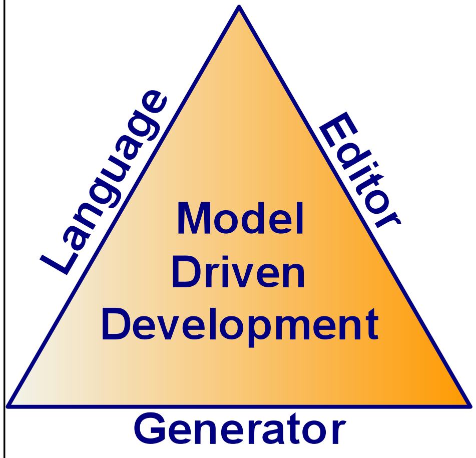
Angel Roman

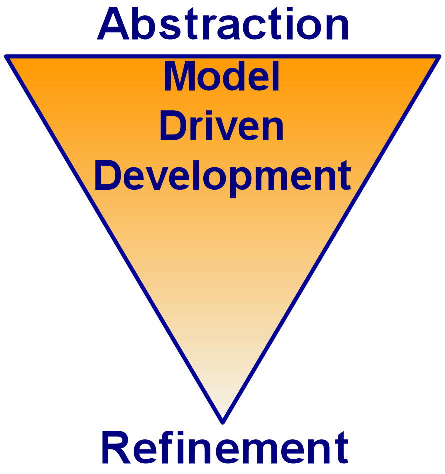
MDE Systems

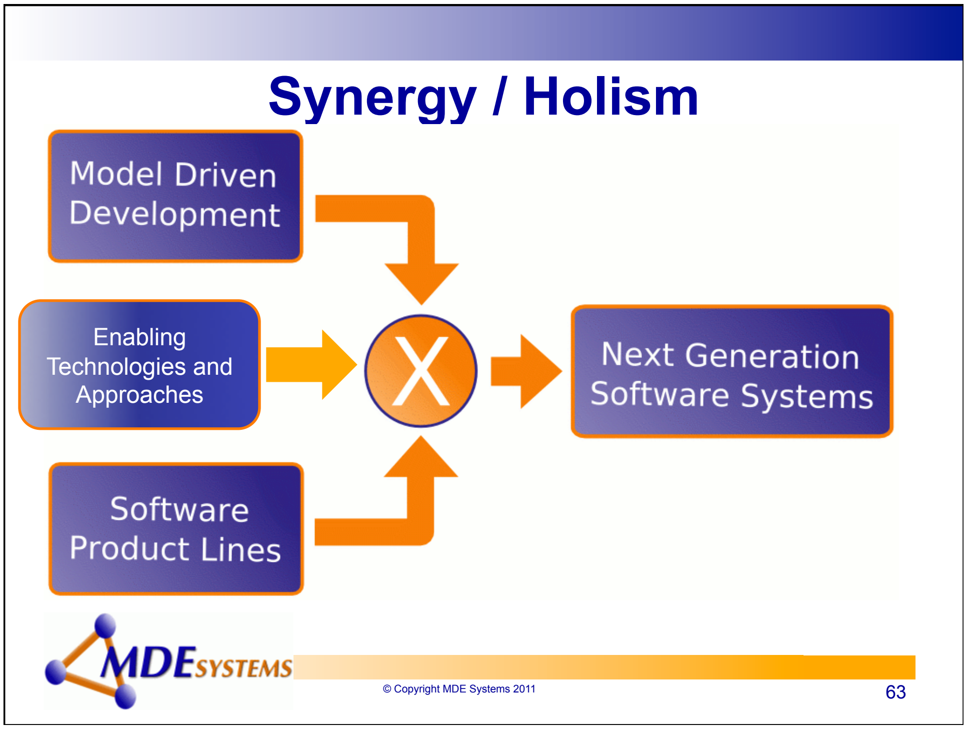


SPLC - Vol. II, September 02 - 07 2012, Salvador, Brazil

Summary







Synergy / Holism

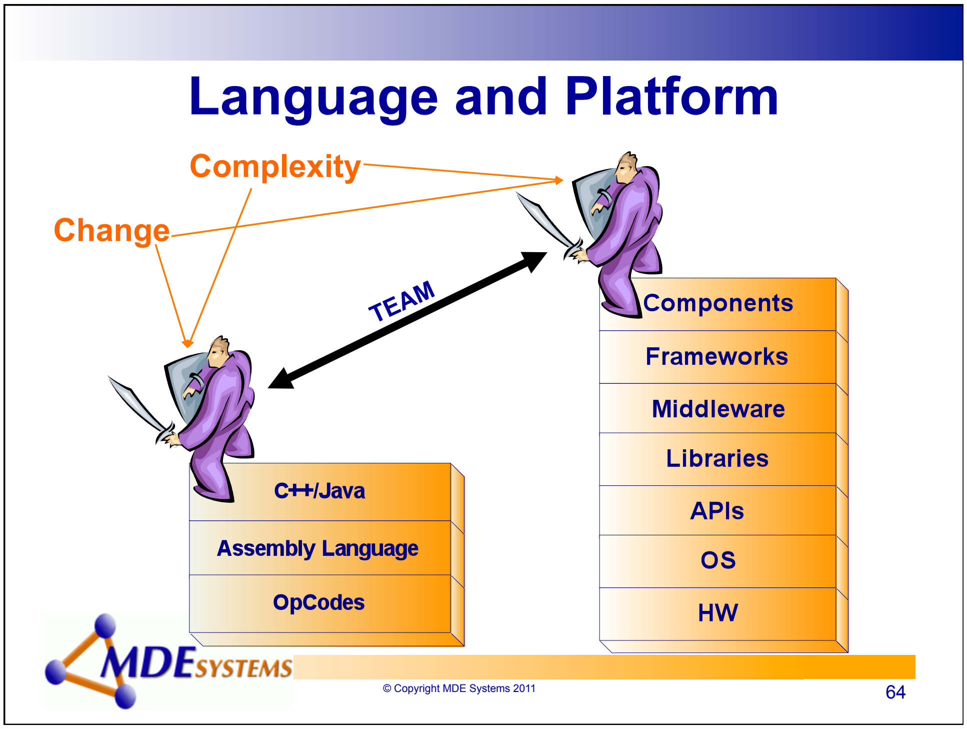
Model Driven Development

Enabling Technologies and Approaches

Software Product Lines

X

Next Generation Software Systems



Language and Platform

Complexity

Change

TEAM

Components

Frameworks

Middleware

Libraries

APIs

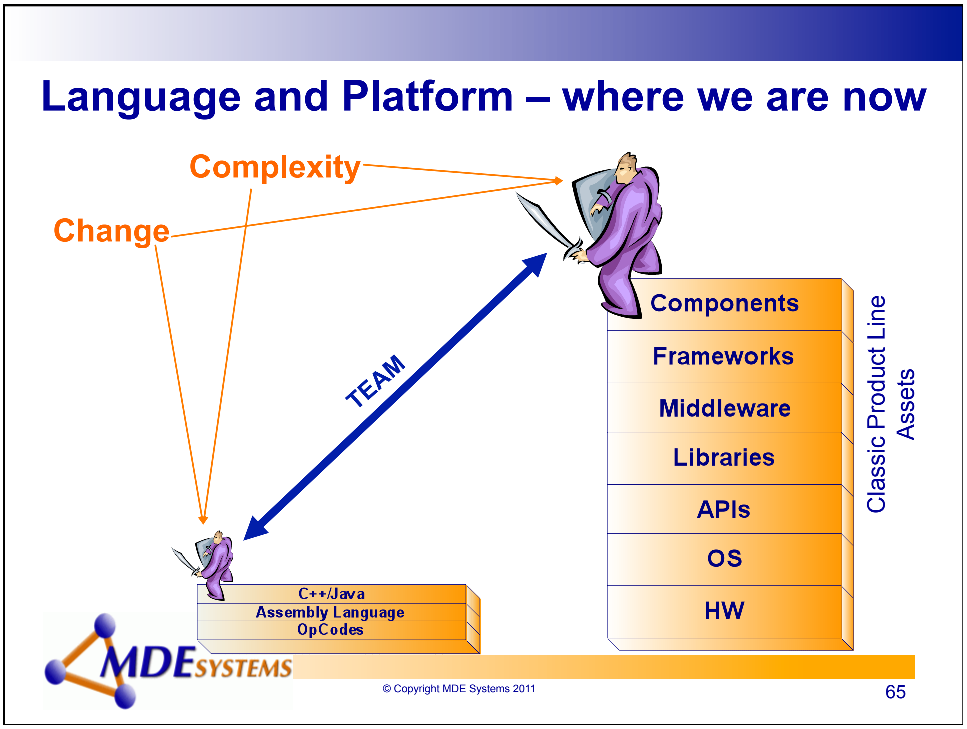
OS

HW

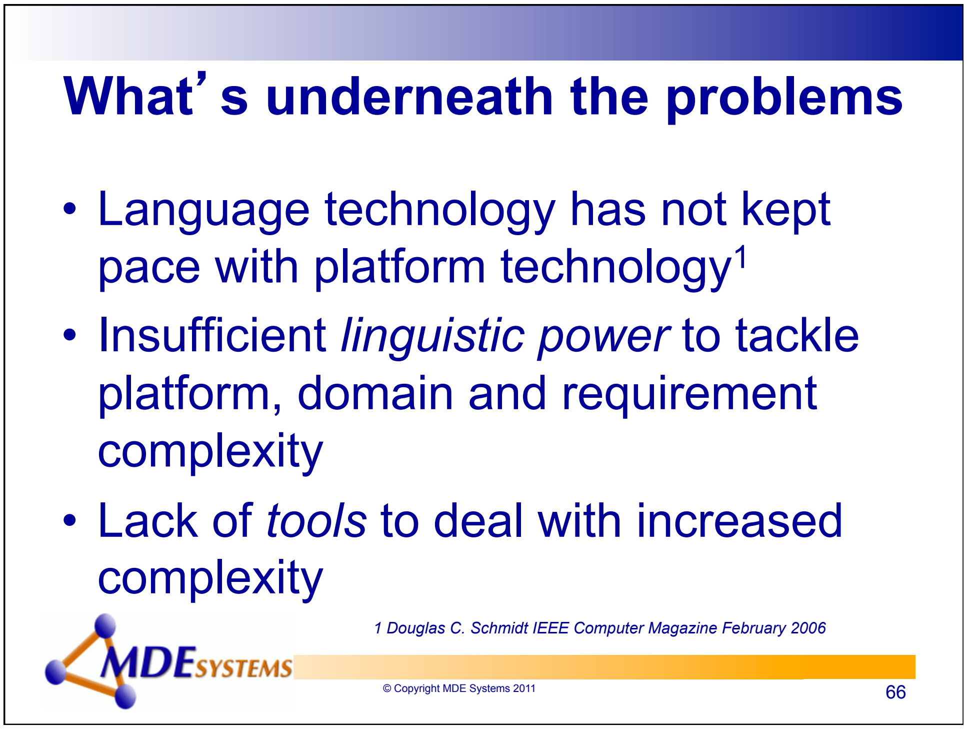
C++/Java

Assembly Language

OpCodes



Language and Platform – where we are now



What's underneath the problems

- Language technology has not kept pace with platform technology[1]
- Insufficient *linguistic power* to tackle platform, domain and requirement complexity
- Lack of *tools* to deal with increased complexity

1 Douglas C. Schmidt IEEE Computer Magazine February 2006

Solution

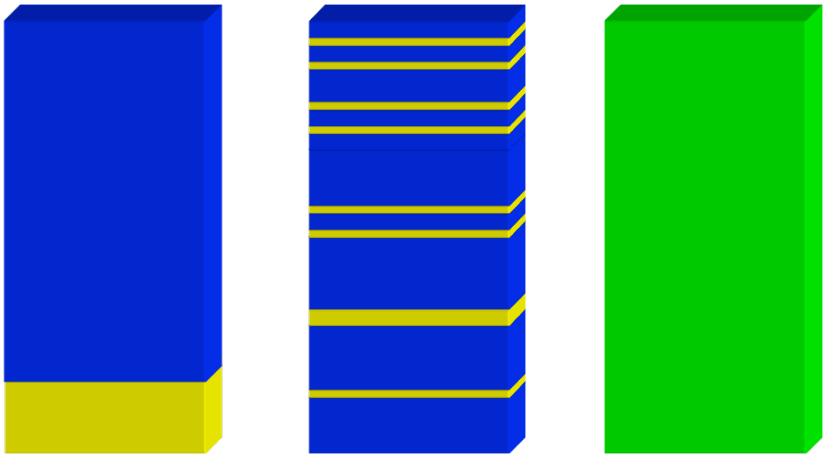
- Leverage recent critical innovations to provide a quantum leap of language technology and tools to overcome the complexity gap

---

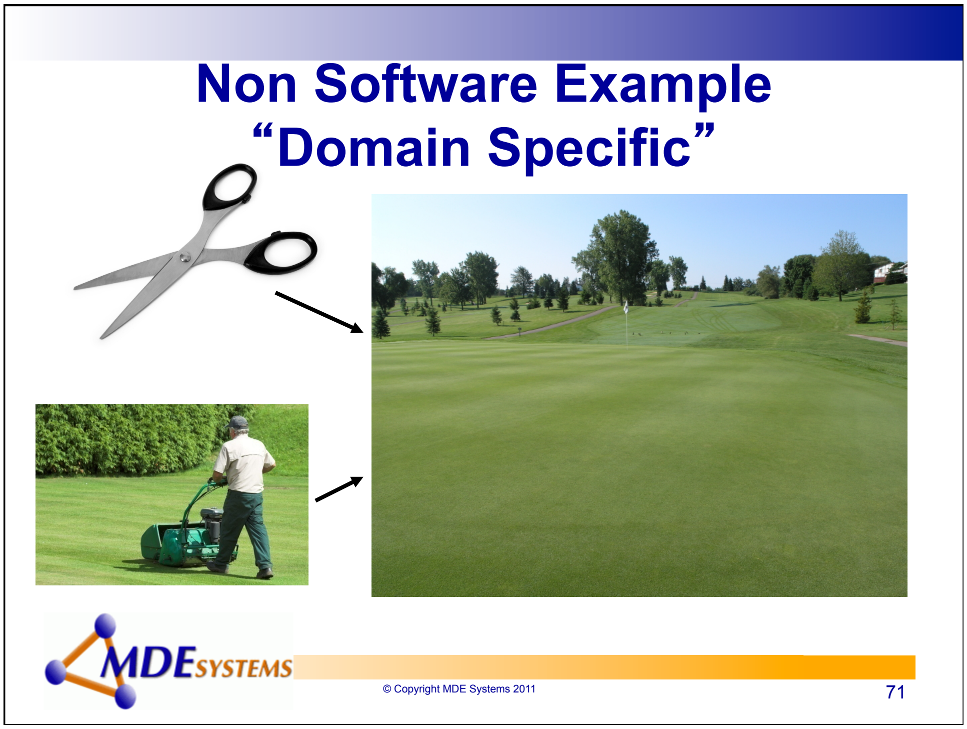
Orders of Magnitude

- 1000x more processing power
- 1000x more dynamic memory
- 1000x more disk space
- 1000x more power efficiency
- 1000x smaller
- 15 orders of magnitude

Rote vs. Creative Code



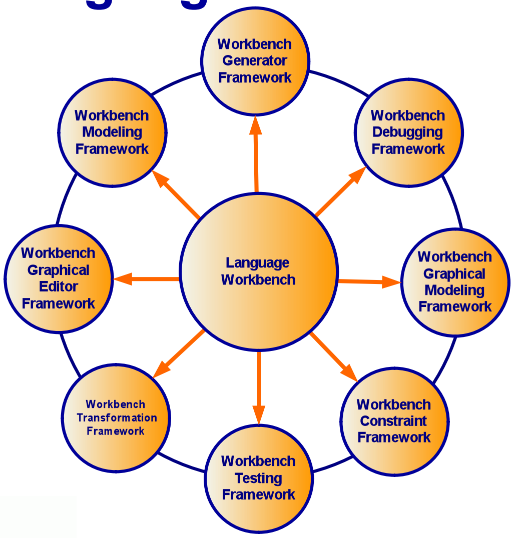
Non Software Example "Domain Independent"



Non Software Example “Domain Specific”

Non Software Example “Domain Specific”

Language Workbench



Language Workbench

Workbench Generator Framework

Workbench Debugging Framework

Workbench Graphical Modeling Framework

Workbench Constraint Framework

Workbench Testing Framework

Workbench Transformation Framework

Workbench Graphical Editor Framework

Workbench Modeling Framework

Questions for any of our tutorial presenters?