Underpinning a Method Engineering Framework with a Powertype-based Metamodel – the FAME Project

3rd AgentLink III Technical Forum meeting (AL3-TF3)

Budapest, September 15-17, 2005

Brian Henderson-Sellers
Director, COTAR
University of Technology, Sydney
www-staff.it.uts.edu.au/~brian
email: brian@it.uts.edu.au

Preview

- I. The FAME project
- II. Method engineering
- III. Metamodelling
- IV. Incorporating a standard method metamodel
- V. Existing repository
- VI. FAML overview
- VII. In summary

I. The FAME Project

(FAME = Framework for Agent-oriented Method Engineering)

Project funded by Australian Research Council (2004-6)

Lead researchers: Brian Henderson-Sellers, Graham Low

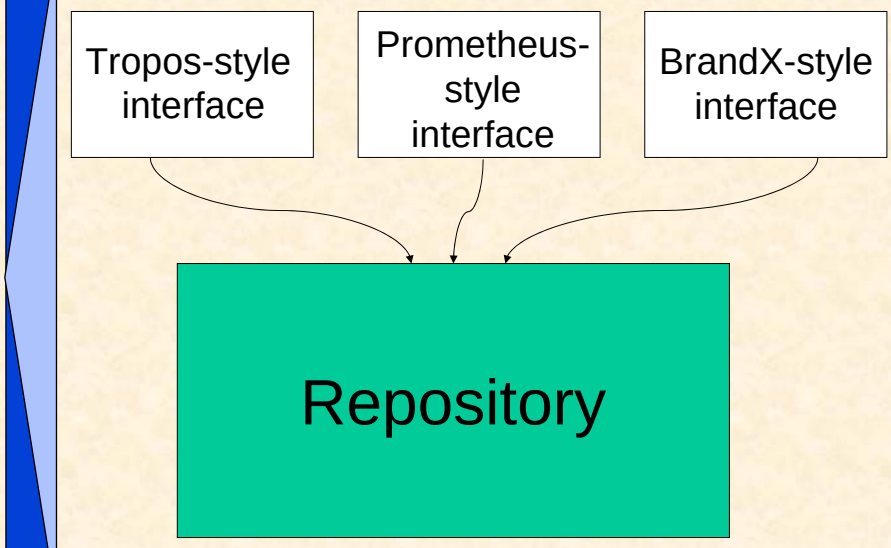
Postdoc researchers: Cesar Gonzalez-Perez, Ghassan Beydoun

Project to

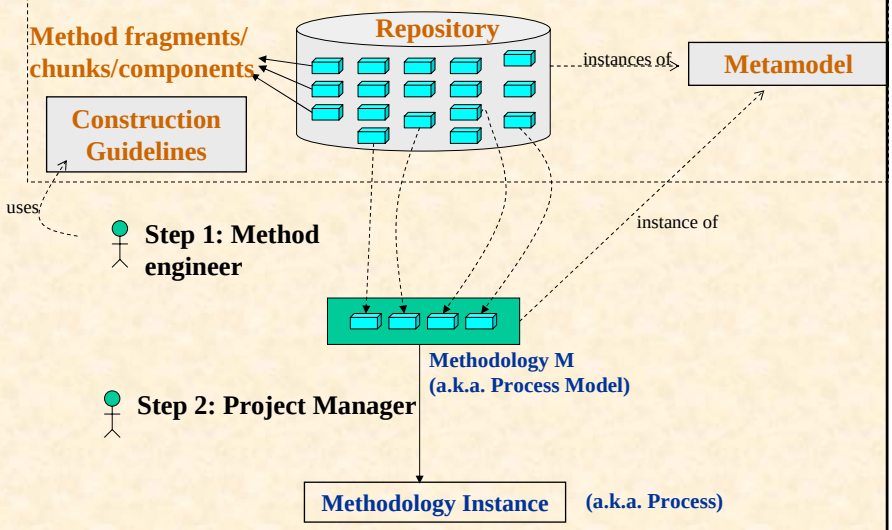
- create an AO, method engineering (ME)-based approach to software development
- Offer a supportive and integrative framework to consolidate and strengthen existing AO methodologies
- FAME project includes both process and product aspects (based on AS4651 and forthcoming ISO standard) including an AO modelling language (FAML) based on a generic model of agents at both design and run time

We thus seek consensus, whilst ensuring consistency and maximizing coverage

- We seek collaborative incorporation of fragments from all other identified AO methodologies
- We propose continuing to support these various methodologies by providing a set of interfaces (façades) to the repository to maintain consistency for current AO methodology users

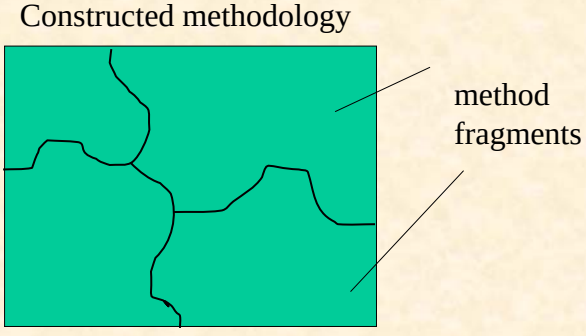


II. Method Engineering



 7

From the method fragments in the repository can be assembled an individually tailored process

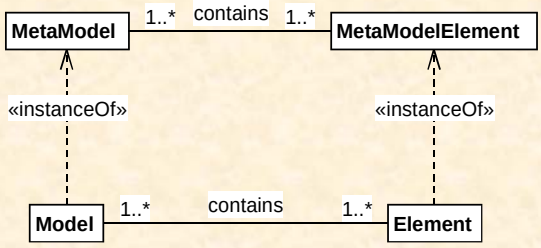


 8

III. Metamodelling

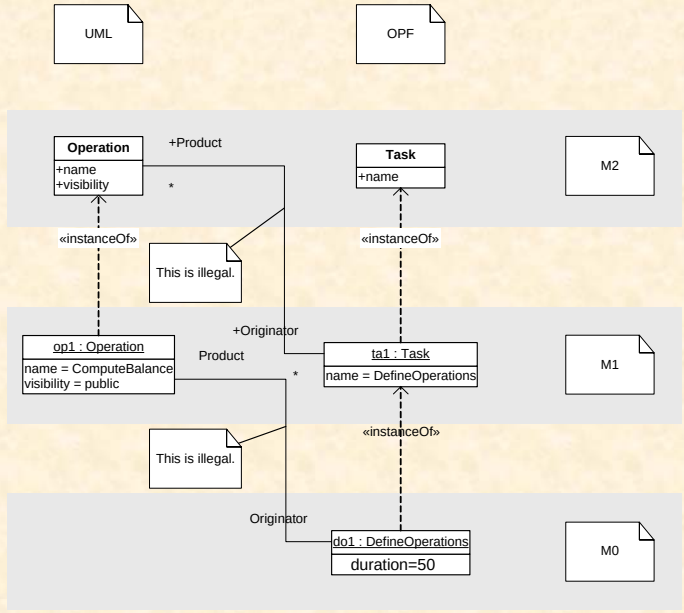
A metamodel is at a higher level of abstraction than a conventional model. It is often called “a model of a model“. It provides the rules/grammar for the modelling language (ML) itself. The ML consists of instances of concepts in the metamodel.

Strict Metamodelling



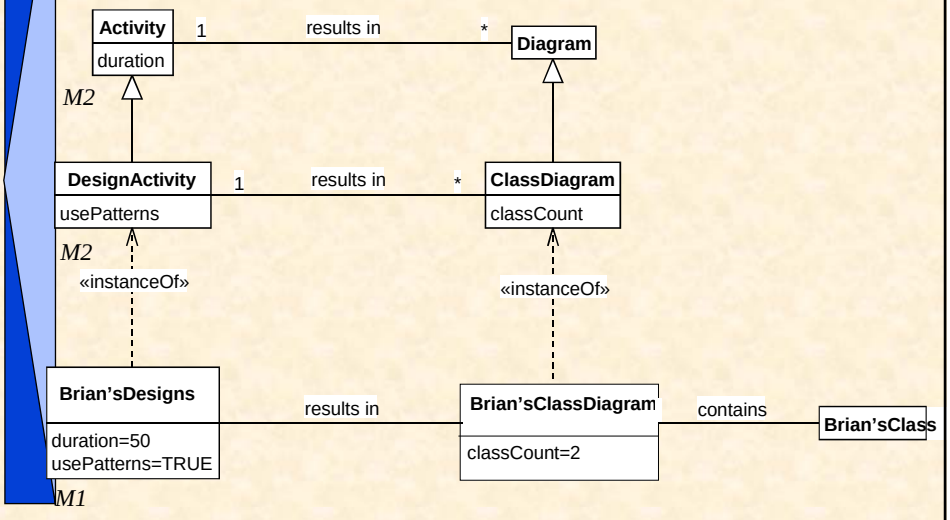
“is-instance-of“ is key relationship i.e.
instance -> class is paralleled by
element -> set
BUT “is-instance-of” is not transitive

Adding process to product adds problems



Enacted tasks need to have a duration

An apparent solution using generalization



In this "solution"

- element Activity can now define an attribute duration.
- Brian'sClassDiagram and Brian'sClass at same level (M1)

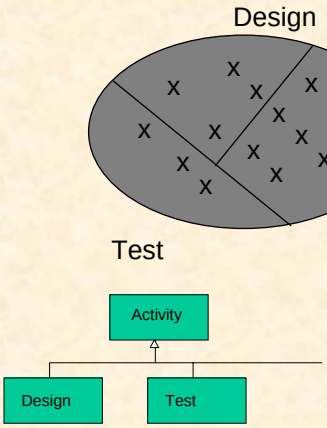
BUT Have lost processes being enacted at M0 and *not* M1

AND M2 level standardization has to identify *all* Activities, all Tasks etc. i.e. *all* contents of a method fragment repository

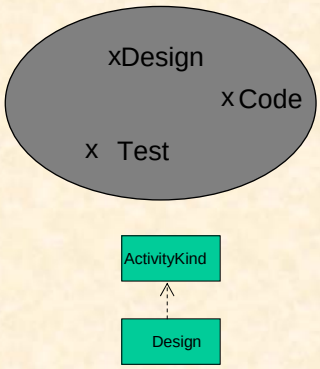
FURTHERMORE

Semantics of "Activity" have been completely changed. [This is a commonly occurring error in the metamodelling literature]

Activity class



ActivityKind class



So, in "Strictness restored" slide, we have also changed the original "Activity" to "ActivityKind" but forgotten to rename it as such. ActivityKind and Activity are two very different Sets. Here Activity class has 14 elements, ActivityKind class has only 3.

IV. Incorporating a standard method metamodel

Current possibilities include

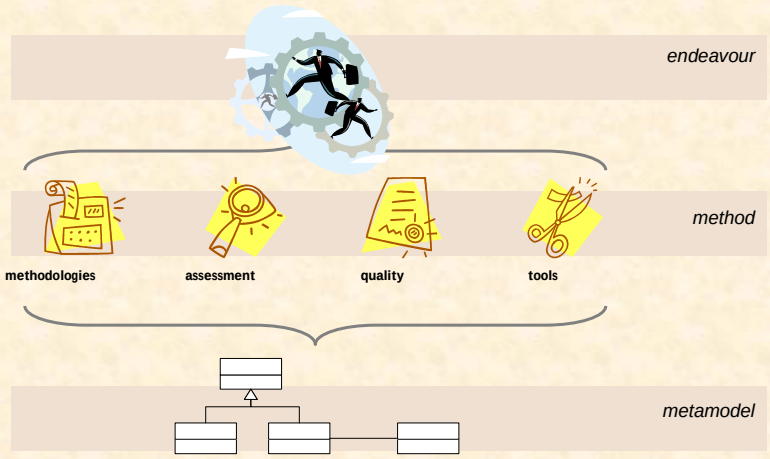
- OMG's SPEM
- AS4651 (SMSDM)/draft of ISO24744 (SEMDM) – used here

SMSDM/SEMDM

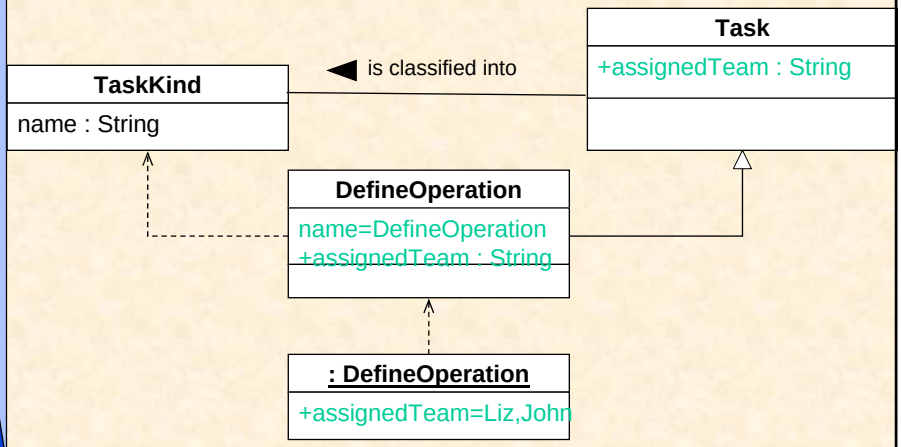
Standard Metamodel for Software Development Methodologies (AS4651-2004 standard)/Software Engineering – Metamodel for Development Methodologies (draft ISO24744)

- Underpinned by powertype patterns
- Three layer architecture: metamodel, method, endeavour

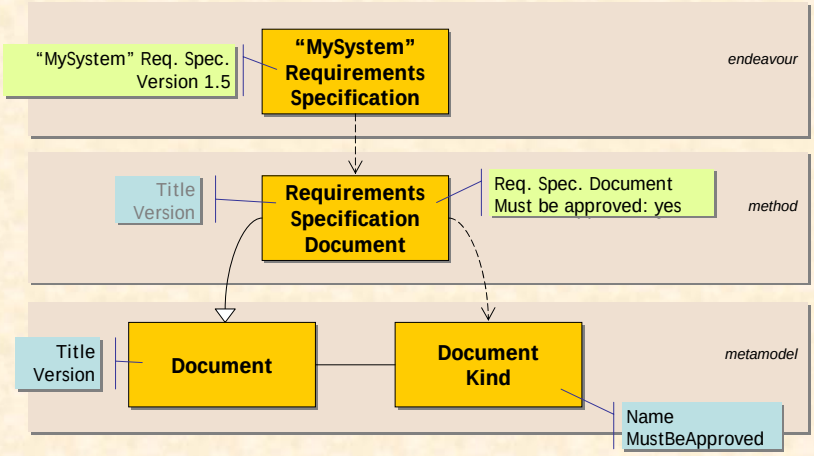
SMSDM/SEMDM architecture



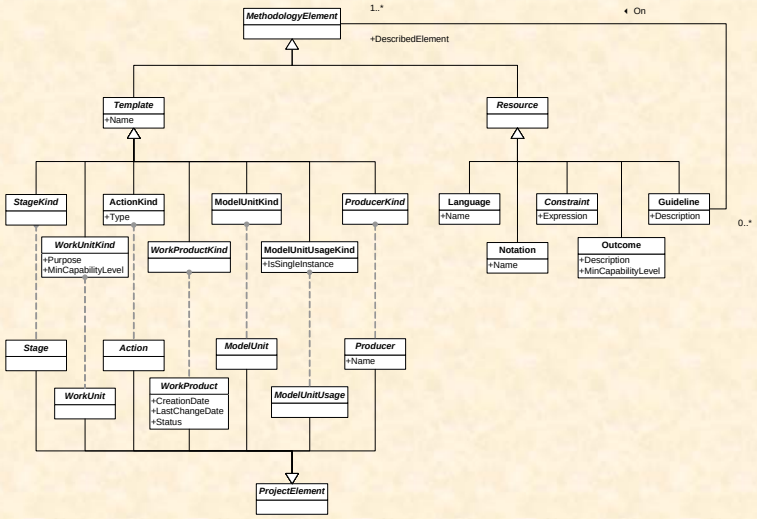
An Example of a Powertype in Process Modelling



Solves the problem of non-transitivity



In summary: the core of the SMSDM/SEMDM



V. Existing Repository

- Precursor to FAME project focussed on the OPF repository
- Fragments consistent with OPF metamodel are currently being (easily) translated to be SEMDM-compatible
- Existing fragments offer wide software development support beyond existing AO methodologies

AOSE fragments

From the literature, we have evaluated Tropos, Prometheus, MaSE, Gaia, Cassiopeia, MAS-CommonKADS, AgentFactory, CAMLE and PASSI for new method fragments

We have so far identified 1 new Activity, 28 new Tasks, 11 new Subtasks, 23 new Techniques and 28 new Work Products

It is now possible to

a) recreate standard AO methods like Gaia, Prometheus

b) create an enhanced or integrated method e.g. Prometheus enhanced by Tropos

Prometheus enhanced by Tropos

	Tasks					
Technique	1	2	3	4	5	6
Abstract class identification						
Agent internal design			Y			
AND/OR decomposition	Y					
Class naming	Y	Y				
Control architecture		Y				
Context modelling	Y			Y		
Delegation analysis	Y	Y				
Event modelling				Y		
Intelligent agent identification		Y				
Means-end analysis	Y					
Role modelling	Y	Y			Y	Y
State modelling		Y				
Textual analysis	Y	Y				
3-layer BDI model		Y	Y			

Key:
1. Model dependencies for actors and goals; 2. Construct the agent model;
3. Design agent internal structure; 4. Model the agent's environment;
5. Model responsibilities; 6. Model permissions

Work Product	Tasks						
	1	2	3	4	5	6	7
Agent Class Descriptor				Y			
Agent Acquaintance Diagram		Y					
Agent Overview Diagram					Y		
Capability Diagram				Y	Y		
Role Model						Y	Y
Role Schema						Y	Y
(Tropos) Goal Diagram	Y						
(Tropos) Actor Diagram	Y						
UML Sequence Diagram		Y		Y			

Key:

1. Model dependencies for actors and goals; 2. Construct the agent model;
3. Design agent internal structure; 4. Model the agent's environment;
5. Model responsibilities; 6. Model permissions; 7. Code

VI. FAML Overview

Start with core concepts of agent:

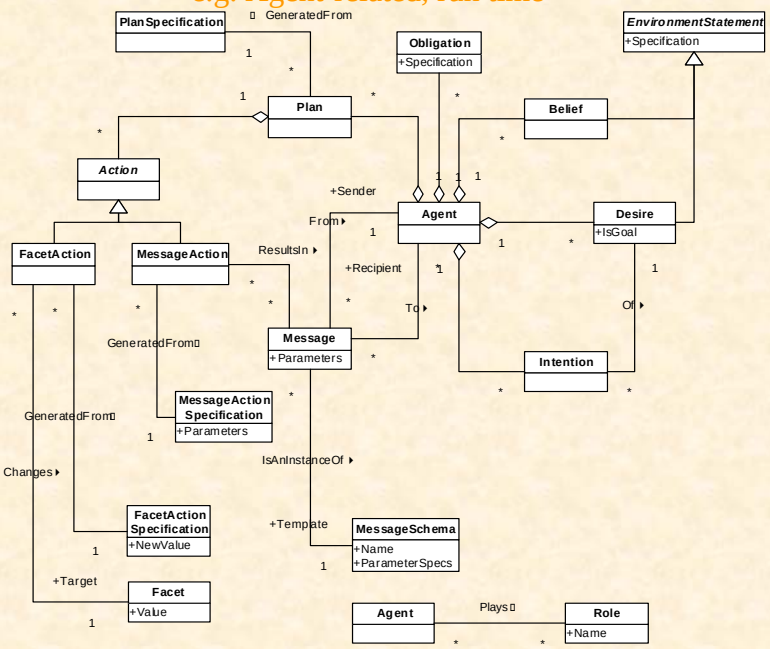
- Autonomy
- Situatedness
- Interactivity

Two scopes

Two layers

	Agent-external	Agent-internal
Design time	System-level	Agent definition-level
Run time	Environment-level	Agent-level

e.g. Agent-related, run time



VII. In Summary

- No one methodology can fit all situations; hence need to create flexibility such that the process remains “standard” yet can somehow be moulded to various circumstances
- Method engineering a solid basis for both standardization and flexibility
- Comprehensive metamodel needed to support process+product aspects of an AO methodology. Simple combination of method metamodels dangerous because of implicit assumptions (e.g. agents collaborate vs. agents compete) and use of same term but with different semantics

In Summary – cont.

- Start with existing repository of method fragments and consolidate
- Implement the new standard metamodel
- Create exemplar methodologies for industry testing
- Encourage community effort to intercompare approaches and make recommendations (1, 2 or more?)
- Identification of weak points for further research endeavours?

THE END