

Model-driven software engineering: Engineering or not?

John Grundy

Professor of Software Engineering

Dean, Software & Electrical Engineering

Director, Centre Computing & Engineering

Software Systems

SWIN
BUR
* NE *

SWINBURNE
UNIVERSITY OF
TECHNOLOGY

Swinburne

▶ think **forward**



- What characterises “Engineering”?
- What is model-driven software engineering?
- Some Examples
- Is MDE really “Engineering”??
- Future directions

What is “Engineering”?



- Wikipedia:

“Engineering is the discipline, art and profession of acquiring and applying technical, scientific, and mathematical knowledge to design and implement materials, structures, machines, devices, systems, and processes that safely realize a desired objective or invention.”

- ABET:

“... creative application of scientific principles to design or develop structures, machines, apparatus, or manufacturing processes, or works utilizing them singly or in combination; or to construct or operate the same with full cognizance of their design; or to forecast their behavior under specific operating conditions; all as respects an intended function, economics of operation and safety to life and property”



- Problem needing solving
- Requirements & Specifications
- Designs – usually multiple options; function & aesthetics & safety & ...
- Mathematical models
- Scientific theories/principles underlying models
- Analysis of models to predict outcomes of different options
- Computer-aided design, manufacturing, analysis, processes
- Repeatable processes, project management principles
- Sharing of best practices, professionalism, ethical behaviors, ...



- No “physical” models to ground, constrain, inform
- Artefacts highly changeable through engineering lifecycle [Note: just because CAN change, doesn’t mean SHOULD!]
- What are appropriate models/modelling languages?
- Where does “design” end and “construction” begin?
- Are our solutions/processes repeatable?
- Can we evaluate results – before and/or after construction?
- Can (and do) we capture “best practice”?
- Widespread practices of professionalism and ethics?



- Wikipedia 😊:

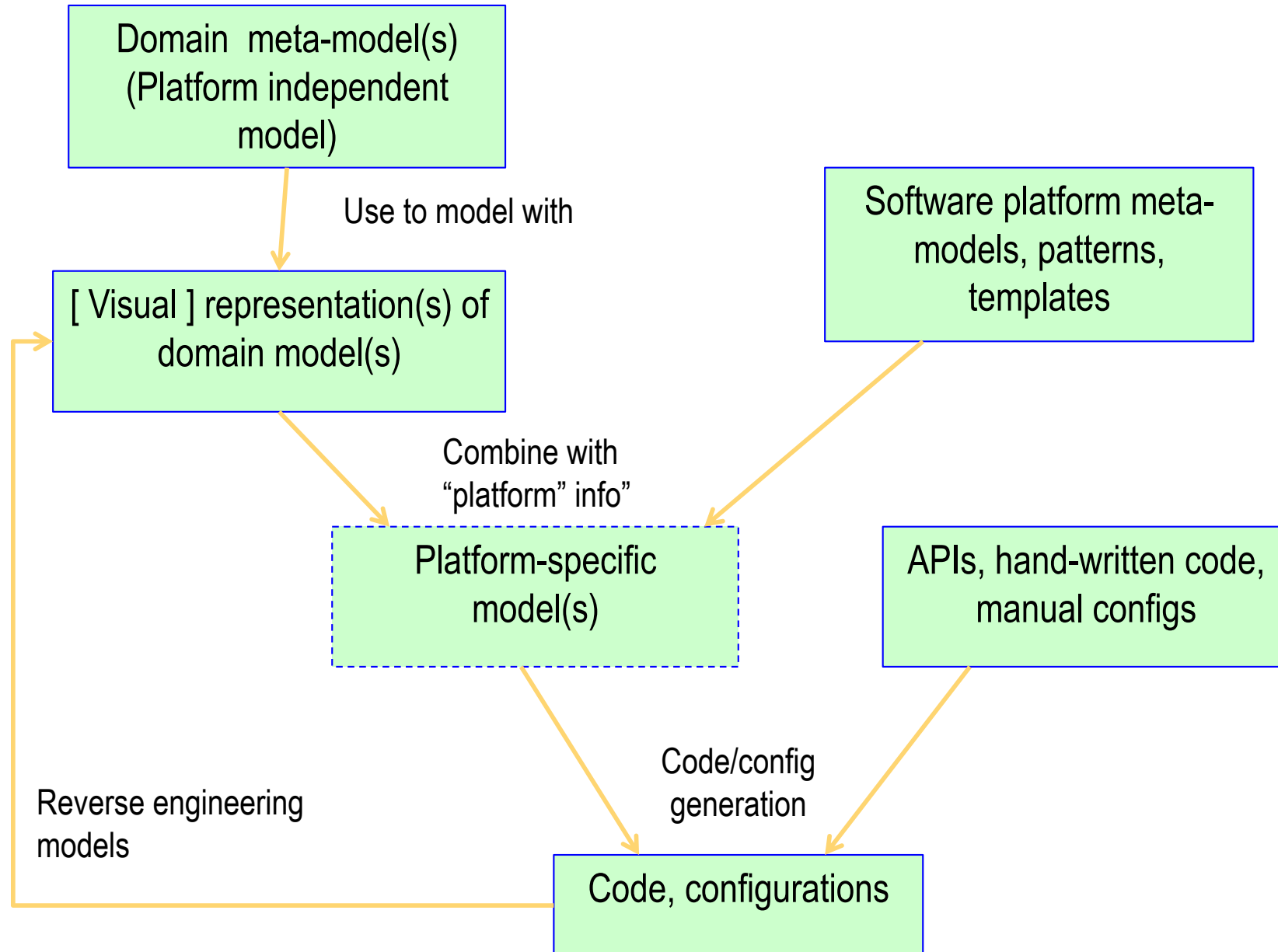
MDE is a software development methodology which focuses on creating models, or abstractions, more close to some particular domain concepts rather than computing (or algorithmic) concepts. It is meant to increase productivity by maximizing compatibility between systems, simplifying the process of design, and promoting communication between individuals and teams working on the system.

- Doug Schmidt, COMPUTER Guest Intro on MDE:

Model-driven engineering technologies offer a promising approach to address the inability of third-generation languages to alleviate the complexity of platforms and express domain concepts effectively.



- Programming languages (3GLs) too low-level to describe many abstractions in software engineering
- SE models too disconnected from 3GLs (program code) e.g. traditional analysis & design languages
- BUT: such models CAN be used to “construct” software directly
- Need high-level modelling languages to better express requirements, architectures, designs, tests etc BUT that can be directly turned into/related to code constructs
- Need to provide ways to build, reason with models, translate models to(/from) code
- BUT: working directly with code (3GLs) still very useful!!!





- Domain meta-model(s), models
- Visualisation(s) of domain models – textual and graphical
- Mapping between models
- Editing tools for models
- Transformation support i.e. model->model, model -> code
- Visualisation support e.g. code/data -> model
- Reasoning support e.g. analysis of models – completeness, correctness, consistency
- Model management support e.g. version control, diffing/merging, etc



- MaramaMTE (Performance Engineering)
- Form-based Mapper (Complex Data Mapping)
- MaramaVCPML (Personal Care Plan App generator)
- MaramaEML (Enterprise Modelling Language)
- MaramaAI (Requirements capture)
- VikiBuilder (Visual Wiki generation)



- Performance test generator from high-level architectural models
- Domain models = architecture, usage models
- Platform models = web domain, multi-tier domain, J2EE, .NET architectures and APIs
- Code, configs = Java, C#, JMeter, MS ACT, Selenium
- Visualisation = performance data on architecture data
- With CSIRO and several small-to-medium companies



The screenshot displays a multi-windowed Java IDE environment. The top-left window shows a UML class diagram for a video system architecture. The top-right window shows a JMeter test plan XML configuration. The bottom-left window shows a Microsoft Excel spreadsheet with performance data. The bottom-right window shows a 3D bar chart comparing performance across three runs.

UML Diagram (1): A class diagram showing a `ClientTest` class with methods `findVideo`, `rentVideo`, `findCustomer`, and `updateCustomer`. It interacts with a `VideoManager` class via `req_1` through `req_4`. The `VideoManager` class has methods `service_1` and `service_2`. A `ToolForApp Server` is also shown.

JMeter Test Plan: A JMeter test plan configuration for version 1.2. It includes a `TestPlan` with properties for user-defined classpath, comments, functional mode, and thread groups.

Excel Spreadsheet (2): A spreadsheet titled `corba-java.txt` with the following data:

method	total	calls	percall
findVideo	4120	100	41.2
rentVideo	6100	40	152.5
findCustomer	1370	60	22.83333
updateCustomer	220	10	22

3D Bar Chart (3): A 3D bar chart titled `comparison 7` showing performance in milliseconds for three runs (total run1, total run2, total run3) across four client requests: `findVideo`, `rentVideo`, `findCustomer`, and `updateCustomer`.



(a) State transition diagram showing various service tasks like Signin, Comm_GET, Data_SELECT, GetIndex, GetProductDetail, Checkout, etc., with transition probabilities.

(b) State transition diagram showing a sequence of tasks: StartPage, GET, GETResult, EndService.

(c) XML configuration snippet for a task:

```
<Path PathID="0">
  <GeneralUserClient UserClientID="0" UserClientName="Client">
    <ClientState>false</ClientState>
  </GeneralUserClient>
  <StartPoint StartPointID="0" StartPoint="Start">
    <IsStartState>false</IsStartState>
  </StartPoint>
  <Task TaskID="0" TaskName="Comm_GET">
    <ParentTask TaskID="0" TaskName="Comm_GET">
      <ResourceLocation>http://136.186.6.235:8080</ResourceLocation>
    </ParentTask>
    <Path PathID="0">
      <CommunicationServiceClient ServiceClientID="0" ServiceClientName="Comm_GET">
        <IsMultiThread>false</IsMultiThread>
      </CommunicationServiceClient>
    </Path>
  </Task>
</Path>
```

(d) XML configuration snippet for a VMGroup:

```
<VMGroup VMGroupID="0" VMGroupName="VMGroup">
  <NumberOfVM>1</NumberOfVM>
  <HostServiceLocation>https://136.186.6.60/sdk</HostServiceLocation>
  <SourceElementConnected>VMManagement</SourceElementConnected>
  <VirtualMachine VMID="0" VMName="FEI_VM1">
    <CacheOfVirtualProcessor>N/A</CacheOfVirtualProcessor>
    <ProcessorDescription>null</ProcessorDescription>
    <RAMDescription>null</RAMDescription>
    <FrequencyOfVirtualProcessor>2.8</FrequencyOfVirtualProcessor>
  </VirtualMachine>
</VMGroup>
```

(e) Architecture diagram showing DataCenter, DataCenterController, PhysicalServer, VMManagement, and VMGroups (FEI_VM2, FEI_VM1) with associated tasks like Signin, GetIndex, Checkout, Signin, GetHelp.

(f) LiveGraph interface showing data file settings, graph settings, and viewport controls.

(g) LiveGraph data plot (a) showing Average Disk Usage (KBps) over time.

(h) LiveGraph data plot (b) showing Host Power Consumption (Watts) over time.

(i) LiveGraph data series settings table:

Show	Label	Colour	Transform...	Tran...
<input checked="" type="checkbox"/>	Average Disk Usage(KBps)	Blue	Actual value	1
<input checked="" type="checkbox"/>	Average Disk Read Rate(KBps)	Purple	Actual value	1
<input checked="" type="checkbox"/>	Average Disk Write Rate(KBps)	Red	Actual value	1
<input checked="" type="checkbox"/>	Data Stroe read request(number/s)	Yellow	Actual value	1
<input checked="" type="checkbox"/>	Data Stroe write request(number/s)	Green	Actual value	1
<input checked="" type="checkbox"/>	Network Packets Received(number/s)	Cyan	Actual value	1
<input checked="" type="checkbox"/>	Nekwork Packets Transmitted(number/s)	Light Blue	Actual value	1

(j) LiveGraph data series settings table:

Show	Label	Colour	Transform...	Transform...
<input checked="" type="checkbox"/>	Host Power Consumption(Watts)	Red	Actual value	1



- Scenario: complex XML or EDI message format; want to translate into a different format; many to process
- Traditionally: write QVT/ATL/XSLT/code to do
- Alternative: model transformation visually and generate these transformation implementations
- Meta-model = source/target and mappings
- Visual models might include forms, trees, concrete data visualisations
- MDE = generate XSLT, ATL, Code (C++, Java),...
- Done various with Orion Health Ltd, XSOL Ltd, NICTA



The screenshot displays the Mapper V1.0 interface, which is used for defining data mappings between source and target data forms. The interface is divided into several sections:

- Source Data Form (Left):** Contains fields for a person (id: 1234, name: Grundy, John, email: john-g@cs.auckland.ac.nz, href: www.cs.auckland.ac.nz/~john-g) and an order (date: 20th March 2002, item: book: How to use Java, qty: 1, price: \$49.95).
- Target Data Form (Right):** Contains fields for an order (date: 20/03/02, total_price: 49.95, customer_info: name: John Grundy, address:) and an item (book_info: How to use Java, quantity: 1, total_cost: 49.95).
- Mapping Rules (Bottom):** A list of rules defining how data is transformed. Examples include:
 - `orders.order.date = Date(person.orders.order.date,"ddmmyy")`
 - `orders.order.customer_info.name = person.name.given + " " + person.name.family`
 - Rules for `minOccurs=0`, `maxOccurs=inf`, and `type=string`.

Red arrows indicate the mapping of data from the source form to the target form. For example, the date field in the source order is mapped to the date field in the target order, and the book title and price are mapped to the book_info and total_cost fields in the target item.



Visualiser | Mapper | Skin Designer

File Tools

Source Visualisation

New Green Building

Living Area	Upper Rooms	Third Floor Rooms
Open Kitchen Kitchen	Room 2 BedRoom	Room 5 BedRoom
Geometry Name Type ing Area Toilet	Room 3 BedRoom	Room 6 BedRoom
Room 1 BedRoom	Room 4 BedRoom	

Target Visualisation

CityCouncil

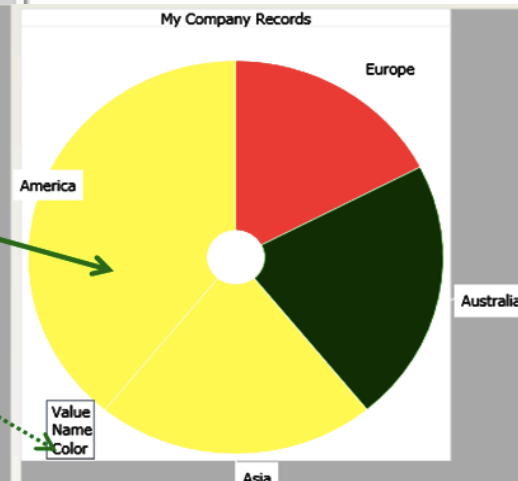
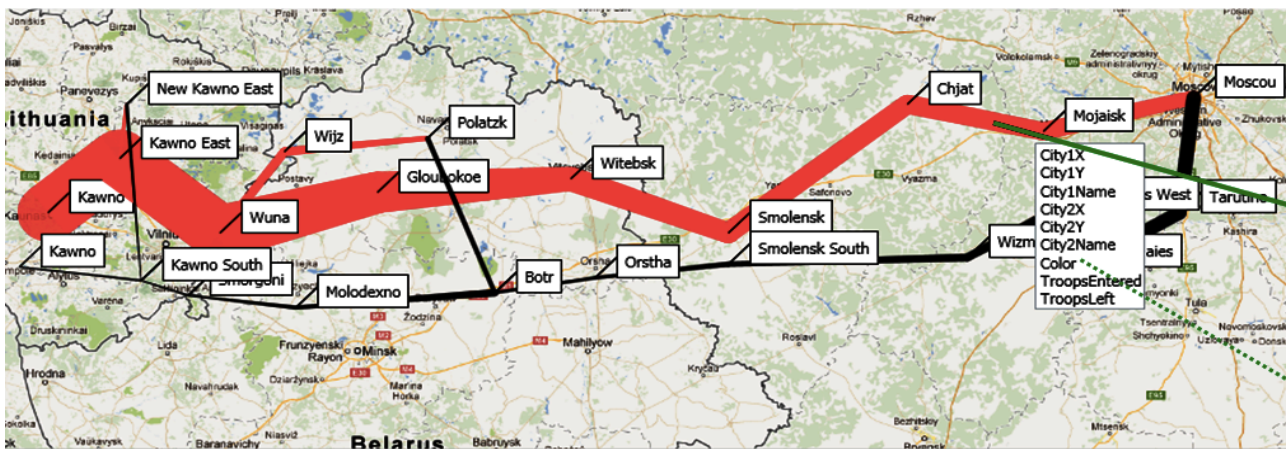
Ground	First Floor	Second Floor
Toilet	Toilet	201
G1	Kitchen	Toilet
Name	S104	202
Color	101	203
Stock1	102	204
	103	205

Mapping Functions

Mapping Rules

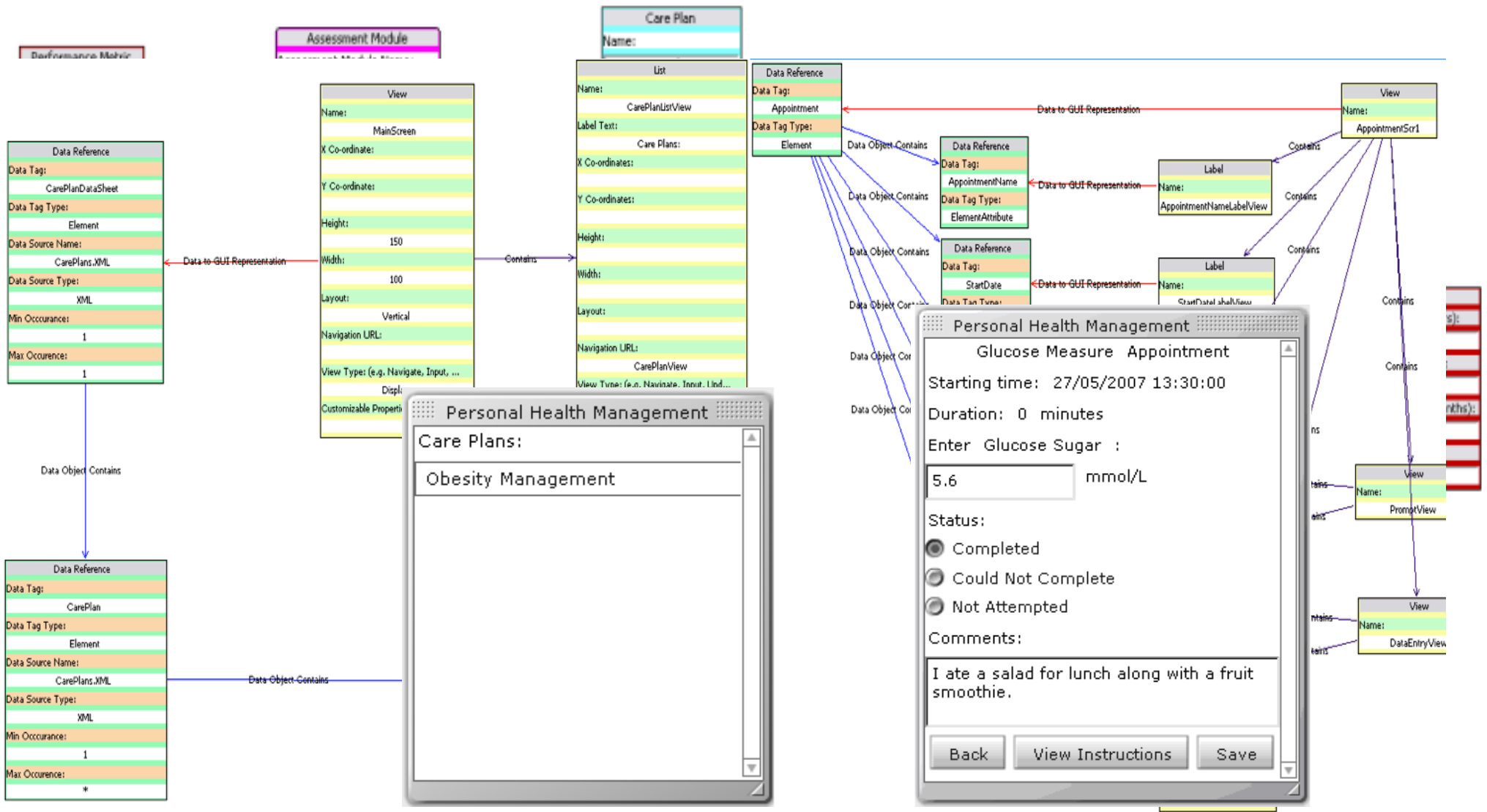
KKitchen Kitchen ↔ room1

Figurative Map of successive losses in men of the French army in Russian Campaign 1812 ~ 1813





- Mobile phone-based personal health care planning applications
- Two meta-models with associated DVSLs: Visual Health Care Planning Language, Visual Care Application Model
- Model generic care plan with a visual DSVL tool
- Configure generic care plan for individual
- Model mobile app UI for individual from tailored care plan with a visual DSVL tool
- Generate Flash, Windows Mobile, iPhone app code





- Enterprise modelling tool
- Integrated domain meta-model synthesized from several existing & new models (BPML, EML, ViTABaL-WS, ...)
- Multiple views with different DSVLs
- Platform meta-model & “code” = BPEL – executable process modelling language (a DSL)
- Tool support for large scale diagram management
- Tool support for model checking for integrated LTSA tool

This image shows a screenshot of the Eclipse IDE with the LTSA Perspective, displaying a Business Process Model and Notation (BPMN) diagram for a University Enrollment Service. The diagram is divided into four main sections: (a) the main process flow, (b) the XML source code, (c) a sequence diagram, and (d) the compiled sequence diagram.

(a) Main Process Flow: The diagram illustrates the 'University Enrollment Service' process. It starts with a 'Student Service' (green circle) and an 'Enroll a Course' event (blue square). The process flows through several tasks: 'Search Course Database' (P1.1), 'Apply Enrollment' (P1.2), 'Make Payment' (P1.3), 'Modify Enrollment' (P1.4), 'Receive Information' (P1.11), and 'Apply Loan' (P1.12). It then branches into two paths: 'Fail' (C1) leading to 'Receive Information' (P1.11) and 'Apply Loan' (P1.12), and 'Pass' leading to 'Enrollment Office' tasks: 'Receive Application' (P1.4), 'Check Academic Records' (P1.5), 'Check Other Conditions' (P1.7), 'Approve Application' (P1.8), and 'Reject Application' (P1.9). The 'Pass' path also includes 'Report Student Records' (P1.6), 'Approve Application' (P1.6), 'Reject Application' (P1.9), and 'Modify Application' (P1.10). The process continues through 'Department' tasks: 'Receive Application' (P1.5), 'Check Academic Records' (P1.7), 'Approve Application' (P1.8), and 'Reject Application' (P1.9). It then moves to 'Finance Office' tasks: 'Modify Payment' (P1.13), 'Receive Payment' (P1.14), 'Send Invoice' (P1.17), and 'Request Payment' (P1.18). The process ends with 'Confirm with StudyLink' (P1.19) and 'Create StudyLink' (P1.20). The process is terminated by a 'Community' event (yellow circle).

(b) XML Source Code: The XML code defines the 'UniversityEnrollment' process. It starts with a 'receive' event for 'StudentEnquireService' and an 'operation' of 'SelectCourse'. It then includes several 'invoke' activities for tasks like 'CheckEnquires', 'CheckAcademicRecords', 'RecordCheck', 'OtherConditions', 'TutorialTimeImpact', 'LectureTimeImpact', 'PaymentRequest', 'ApplyStudentLoan', and 'ReceiveLoanApplication'.

```

<process name="UniversityEnrollment">
  <sequence>
    <receive portType="StudentEnquireService"
      operation="SelectCourse" variable="Course1
    <invoke name="CheckEnquires" portType="TravelAc
      operation="CheckEnquires" inputVariable="F
    <invoke name="CheckAcademicRecords" portType="U
      operation="CheckAcademicRecoards" inputVar
    <invoke name="CheckAcademic" portType="Departme
      operation="CheckAcademic" inputVariable="S
    <invoke name="RecordCheck" portType="ReportAcac
      operation="Report" inputVariable="CheckEqu
    <invoke name="OtherConditions" portType="OtherC
      operation="ExamTimeImpact" inputVariable="
    <invoke name="TutorialTimeImpact" portType="Imp
      operation="CheckTutorialTime" inputVariabl
    <invoke name="LectureTimeImpact" portType="Chec
      operation="CheckLectureTime" inputVariable=
    <invoke name="PaymentRequest" portType="Finance
      operation="RequestPayment" inputVariable="
    <invoke name="ApplyStudentLoan" portType="Stude
      operation="SendApplication" inputVariable=
    <invoke name="ReceiveLoanApplication" portType=
  </sequence>
</process>

```

(c) Sequence Diagram: A sequence diagram showing a sequence of six messages (0 to 5) between a participant and a machine. The messages are: 0 (red circle), 1 (cyan circle), 2 (cyan circle), 3 (cyan circle), 4 (cyan circle), and 5 (cyan circle).

(d) Compiled Sequence Diagram: A list of compiled sequence diagram elements, including: 'compiled:UNIVERSITYENROLLMENT_RECEIVE1', 'compiled:UNIVERSITYENROLLMENT_CHECKENQUIRES', 'compiled:UNIVERSITYENROLLMENT_CHECKENQUIRES_REPLY', 'compiled:UNIVERSITYENROLLMENT_CHECKENQUIRES_SEQ', 'compiled:UNIVERSITYENROLLMENT_CHECKACADEMICRECORDS', 'compiled:UNIVERSITYENROLLMENT_CHECKACADEMICRECORDS_REPLY', 'compiled:UNIVERSITYENROLLMENT_CHECKACADEMICRECORDS_SEQ', 'compiled:UNIVERSITYENROLLMENT_CHECKACADEMIC', 'compiled:UNIVERSITYENROLLMENT_CHECKACADEMIC_REPLY', and 'compiled:UNIVERSITYENROLLMENT_CHECKACADEMIC_SEQ'.



- Requirements capture and analysis tool
- Textual requirements -> essential interactions -> Essential Use Cases -> UIDs, OOAs
- Domain meta-models include natural language (!), EUCs, UML etc
- Textual and visual representations of domain models
- Transformation of text to/from EUC-based DSVL models
- Analysis of consistency between models, completeness/correctness of models via EUC pattern library



The use case begins when the customer goes to the Customer Log-on page.

user intentions

system responsibilities

The screenshot shows the Eclipse IDE with several windows open. The main window displays two Marama diagrams: 'ListOfAbstractInteraction' and 'EUCDiagram'. The 'ListOfAbstractInteraction' diagram lists several user intentions and system responsibilities. The 'EUCDiagram' diagram shows a sequence of interactions between a user and the system.

A warning dialog box is displayed in the foreground, stating: "Warning: EUC Component sequence is inconsistent with the sequence in the list of abstract interaction." The dialog has "Update" and "Cancel" buttons.

The 'EUC Trace Result' window shows the following sequence of events:

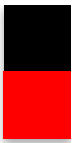
1. Voter loads EVote system is online
2. Voter ***select voter registration*** option
3. EVote system ask for name, social security number
4. Voter provide name and social security number
5. EVote system checks Voter status
6. Evote System generates Voter login id and password
7. 1.a.After 60 sec
 - 1.a. EVote system page
 - 2.a. After 60 sec
 - 2.a.1. EVote system display time out page
 - 3.a After 60 sec
 - 3.a.1 EVote system displays time out page
 - 5.a. Voter data is not in record
 - 5.a.1 Evote System display

The Problems window at the bottom shows the following warning:

Description	Resource	Path	Location
Warnings (1 item)			
Inconsistency: The sequence of EUC component is inconsistent with the of abstract interaction and Textual requirement	diagram1.maramaDiagram	A12	select option

Visual Wikis (“Vickis”) & VickiBuilder

Swinburne



The collage consists of four overlapping screenshots:

- (d) ThinkFree:** A central hub-and-spoke diagram with 'CS9' at the center. It is surrounded by various application icons and categories such as 'Student Administration', 'Grades', 'Applications', 'nDeva', 'Phase 1 SA Upgrade', 'Package', 'uses Technology', 'uses Database', and 'publish Data Exchange'. A sidebar on the right contains text for 'CS9' and 'UoA Application'.
- (a) Thinkbase:** A complex network diagram centered on 'Avatar'. It shows connections to 'Subjects', 'Awards Won', 'Edited by', 'Award Nominations', 'Produced by', 'Executive produced by', 'English Language', 'Gross revenue', 'Enter the World', 'Notable filming locations', 'Wellington Miramar, New Zealand', 'Hamakua Coast', 'Playa Vista', 'Kauai', 'AITrailer', 'Margery Simkin', 'PG-13 (USA)', and 'Avatar'.
- (c) Process Mapper:** A flowchart diagram showing a process starting with 'start Enrollment Process', leading to 'Search course DB', 'Student Service', 'Apply enrollment', 'Apply enrollment', and 'Receive application'. A sidebar on the right contains text for 'Apply enrollment' and 'Student'.
- (b) Thinkpedia:** A large network diagram centered on 'University of Auckland'. It shows connections to various related entities such as 'Auckland College of Education', 'University of New Zealand', 'Auckland University College', 'School of Education', 'City', 'Faculty of Education', 'Race Station', 'Technology', 'Natural Feature', 'Professor', 'Teacher', 'Vice-Chancellor', 'Company', 'Information Management', 'Microsoft Corporation', 'Industry Firm', 'Social Tag', 'Information Management Technologies Corporation', 'Education', 'Association of Commonwealth Universities', 'University of Auckland', 'Academy', 'HEQES World University Rankings', 'Performance Based Research Tun Higher education', 'New Zealand', 'New Mexico', 'United States', 'Leeds West Yorkshire United Kingdom', 'Victoria Seychelles', 'Auckland', 'University of Oxford', 'Province Of State', 'Publish @ Medium', 'Person', 'Organization', 'City campus', 'New Zealand', 'University of Cape Town', 'Auckland University of Technology', and 'Academy'.

Example



The image displays a complex web-based interface for building visual wikis. It is divided into several main sections:

- Left Panel (VikiBuilder v1):** Contains a sidebar with 'Visual Wiki Models' (VW Lostpedia, VW Thinkpedia v2, VW Thinkbase), 'Tools' (New, Run, Run Example, Preview), and a large tree diagram representing the structure of 'VW Lostpedia'. The tree starts with 'VW Lostpedia' at the top, branching into 'MediaWiki read API', 'MediaWiki view access', and 'Semantic query API'. Further sub-nodes include 'unstructured representation', 'MediaWiki representation', 'semantic extractions', 'MediaWiki view', 'graph representation', 'coord id', and 'Thinkmap graph view'. A blue arrow points from the 'VW Lostpedia' node in the tree to the main preview window.
- Top Middle Panel (Freebase):** Shows a search bar and the text 'You're in Edit Mode'. Below it, there's a section for 'Types: VW Visual Wiki' and 'Add another type...'. A blue arrow points from this section towards the tree diagram.
- Right Panel (Preview):** Displays a preview of the 'VW Lostpedia' page. It features a search bar with 'looking glass' entered, a network graph visualization, and a search result for 'The Looking Glass' (Redirected from Lostpedia). A blue arrow points from the search bar area towards the tree diagram.
- Bottom Panel (Detailed Tree):** A zoomed-in view of the tree structure from the left panel, showing nodes like 'unstructured representation', 'MediaWiki representation', 'semantic extractions', 'MediaWiki view', 'graph representation', 'coord x1', 'Thinkmap graph', and 'navigation coordination'. A blue arrow points from this panel towards the main interface.
- Bottom Right Panel (Preview Window):** A separate browser window showing the 'Oceanic Airlines Flight 815' page. It includes a search bar, a network graph, and a text-based article snippet. The article text reads: 'Oceanic Airlines Flight 815 was a scheduled flight from Sydney, Australia to Los Angeles, California, United States, on a Boeing 777 owned by Oceanic Airlines. Under mysterious circumstances, on September 22, 2004, the airliner, carrying 324 passengers, deviated from its original course and disappeared over the Pacific Ocean. This is the central moment in the series Lost and the chronological beginning of the main characters' exploits on the Island. Two months later, wreckage presumed to be Oceanic Flight 815 was found in the Sunda Trench which is in the Indian Ocean near Bali, and all passengers were presumed dead. In reality, however, the discovered wreckage was staged. The real plane had suffered a mid-air break-up and crashed on an uncharted island, with more than 70 passengers surviving the crash itself. Later, six of those survivors made it off the island and became known as the Oceanic Six.' Below the text is a table with details: 'First introduced: Boeing 777', 'Date: 9/22/2004', 'Pilot: Seth Norris', 'Took off from: Sydney', 'Intended destination: Los Angeles', 'Crashed on: the Island'. A blue arrow points from the search bar area of the main interface towards this preview window.



- Higher level models to work with – easier to create, understand, modify, do analysis on than 3GL code
- Can refine models, ultimately to code/configurations
- Repetitive stuff (“construction”) can be largely (sometimes completely) automated

- Models get complex, need good tools to work with them
- Hard to agree on standards esp. in new areas
- Model and tool evolution gets very tricky
- Trade-off between flexibility (code) and productivity (MDE)...

Is MDE “Engineering”?



- Using meta-models, model representations
- Can check (some aspects of) models in tools
- Repeatable processes
- Productivity, quality improvements
- “Construction” becomes push-a-button

- Best practices in MDE?
- Dynamic MDE (change while running)?
- Proactive MDE (change while running in anticipation of problems...)??



- Improve design of meta-models including specify complex constraints
- Improve design and evaluation of DSLs
 - Physics of Notations Tool – new PhD project
- Improve specification of transforms e.g. by using CONVERt ☺
- Improve tool support for DSLs, transforms, “design critics”
 - Horus – our next-generation Web-based DSL tool
- Support other things via MDE e.g. security, HPC, cloud platforms
 - Horus HPC, MDSE @ Runtime, TOSSMA, SMURF
- Proactive adaptation of complex systems via MDE
 - My 2016 DP application... ☺
- Assess MDE on range of real-world problems
- Identify when “best practice” to use, when not to use...

Thank yous

Swinburne



- John Hosking – most of these projects
- Jun Huh, Karen Li – Marama meta-tools
- Rainbow Cai, Feifei Chen – MaramaMTE, StressCloud
- Abizer Khambati – VCPML etc
- Michael Li – Form-based Mapper
- Massila Kalmalrudin – MaramaAI
- Christian Hirsch – VisualWikis, VickiBuilder
- Iman Avazpour – CONVERt
- Mohamed Almorsy – Horus, Horus HPC, MDSE @ R, ...
- Many other contributors over many years...

In case your are interested...

Swinburne



[Marama] Grundy, J.C., Hosking, J.G., Li, N., Huh, J., Ali, M., Li, L. Generating Domain-Specific Visual Language Tools from Abstract Visual Specifications, IEEE Transactions on Software Engineering, 2013

[CONVERT] Avazpour, I., Grundy, J.C., Vu, H. Generating Reusable Visual Notations using Model Transformation, to appear in International Journal of Software Engineering and Knowledge Engineering, 2015.

[VickiBuilder] Hirsch, C., Hosking, J.G. and Grundy, J.C. VickiBuilder: end-user specification and generation of Visual Wikis, In Proceedings of the 25th IEEE/ACM International Conference on Automated Software Engineering (ASE 2010), Antwerp, Belgium, 20-24 Sept 2010, IEEE CS Press.

[MaramaAI] Kalmalrudin, M., Grundy, J.C. and Hosking, J.G. Improving Requirements Quality using Essential Use Case Interaction Patterns, In Proceedings of the 2011 International Conference on Software Engineering (ICSE2011), Honolulu, Hawaii, USA, May 21-28 2011, ACM Press.

[Orion Mapper] Grundy, J.C., Mugridge, W.B., Hosking, J.G. and Kendal, P. Generating EDI Message Translations from Visual Specifications, In Proceedings of the 16th International Conference on Automated Software Engineering, San Diego, 26-29 Nov 2001, IEEE CS Press, pp. 35-42

[Form-based Mapper] Li, Y., Grundy, J.C., Amor, R. and Hosking, J.G. A data mapping specification environment using a concrete business form-based metaphor, In Proceedings of the 2002 International Conference on Human-Centric Computing, IEEE CS Press

[Visual Care Plan Modeller] Khambati, A., Warren, J., Grundy, J., and Hosking, J. Care Planning Systems for Consumer Engagement in Chronic Disease Management, Electronic Journal of Health Informatics, vol 4, no. 1, 2009

[MaramaEML] Li, L., Hosking, J.G. and Grundy, J.C. MaramaEML: An Integrated Multi-View Business Process Modelling Environment with Tree-Overlays, Zoomable Interfaces and Code Generation, Demo session, In Proceedings of the 2008 IEEE/ACM International Conference on Automated Software Engineering, L'Aquila, Italy, 15-19 September 2008, IEEE CS Press.

[MaramaMTE] Cai, Y., Grundy, J.C. and Hosking, J.G. Synthesizing Client Load Models for Performance Engineering via Web Crawling, In Proceedings of the 2007 IEEE/ACM International Conference on Automated Software Engineering (ASE 2007), Atlanta, Nov 5-9 2007, IEEE CS Press

Thanks!



FRST SER SPPI
FRST NERF DS Tools



DP110101340
DP120102653
DP140102185



Malaysian Ministry of Education

