



"OSLO"

# Oslo explained

An early look at Microsoft's  
modeling platform

# Introducing...

- Class-A
  - Kennisprovider
  - Microsoft development
  - Training & Coaching
  - <http://www.class-a.nl>
- Alex Thissen
  - Trainer/ coach
  - <http://blog.alexthissen.nl>



# Agenda

- Microsoft's vision on modeling
- Applications, models, data and runtimes
- Oslo repository
- "M" family of languages
- Tooling in Oslo
- Q&A and discussions

# Popquiz: spot the Oslo model



# Segments of modeling

Oslo focuses on model-driven, but will help for other segments

- Drawings
  - Human to human communication using a model

- Software factories
  - Model-assisted
    - Code generation from a model and model creation from code
    - UML and DSL toolkit

- Oslo
  - Model-driven
    - Direct execution of a model by a runtime

# Microsoft's vision for modeling

- Formalize and generalize ideas to gain productivity
- Enable declarative, model-driven programming
- Put more of your app in data instead of code
  - Unify data into models
  - Use "executable data"
- Create more human readable artifacts for your application
- Involve non-programmers in application lifecycle
- A runtime for each particular domain

# Models in application development

- Declarative models instead of imperative code
  - Express models with data
  - Change model data without touching code
- An application can use multiple models
- Generalized code will behave/adopt/
- Model data drive runtimes
- General purpose models and runtimes are complex (i.e. not for everyone to build)

## Oslo TARGET DOMAINS

Applications

Services

Storage

Processes

People

Deployment

Management

Operations

[Your Domains]

# Oslo domains and runtimes

- Several domains have been modeled
 

Identity & security	ServiceModel and WorkflowModel
Application, management, hosting	Transactions
Language and repository	Messaging
- Oslo helps you define a domain model
- Most domains need or have some runtime
- Microsoft already has a couple of runtimes
  - Workflow service model (WCF+WF 4.0)
  - Various markup engines (HTML, XAML for WPF)
  - Solver Foundation

# Types of data in an application

- Business data
  - Changes often, OLTP data e.g. customer, order
- Application data (drives your application)
  - Configuration
    - Registry or database
    - XML or other text files
  - Reference data
  - Executable data for a consuming runtime

Oslo lets you model these kinds of data



# Data in Domain Specific Languages

DSL gives specialized representation of model data

- Fit for a certain domain
- Targeted at domain experts

Visual design experience  
enables "non-programmers"



Textual DSLs  
for "programmers"



# Inside Oslo platform and SDK

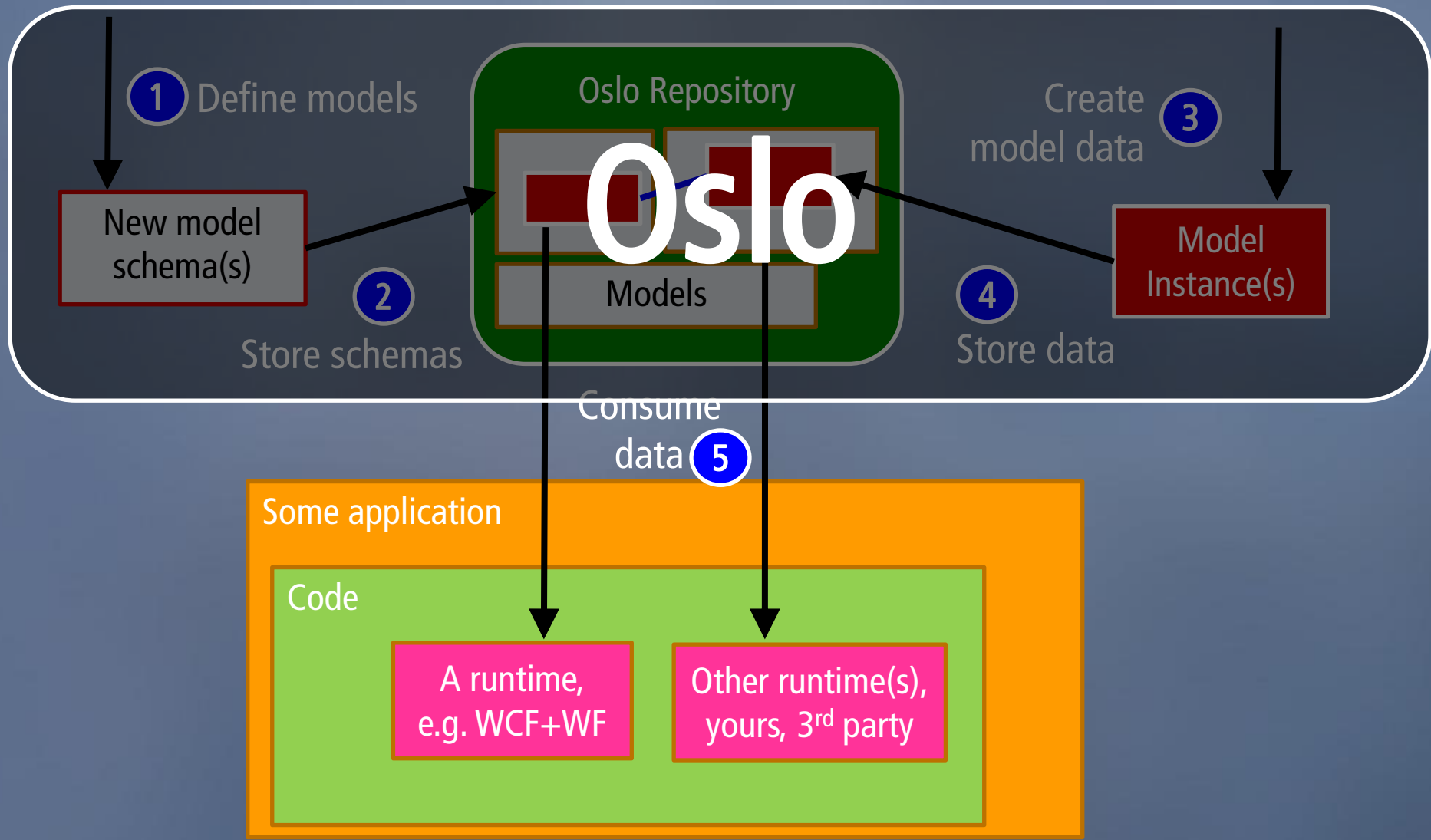


- Repository
  - Store model schemas and instances
- A family of languages
  - Write model schemas, instances and DSLs



- Tools
  - Help you work with models
  - From command-line to new designers to Visual Studio
- Managed API for above

# Oslo application lifecycle



Oslo Mansion: where pretty models live together

# THE REPOSITORY

# Oslo repository

- Storage of model schemas
  - Preloaded with Oslo application domain models
  - Extensible: customize existing or add your own schemas using M
- Storage of model instances
  - Model data for schemas
  - Optimized for many reads and few writes
- Backend storage in SQL Server 2008

# SQL Server as a repository

- Well-known platform
- Very efficient query processor
  - More models == more data
- Use power of SQL Server platform (BI)
  - SSIS for migrating data
  - Reporting
  - Replication and failover clustering
- Authentication and authorization



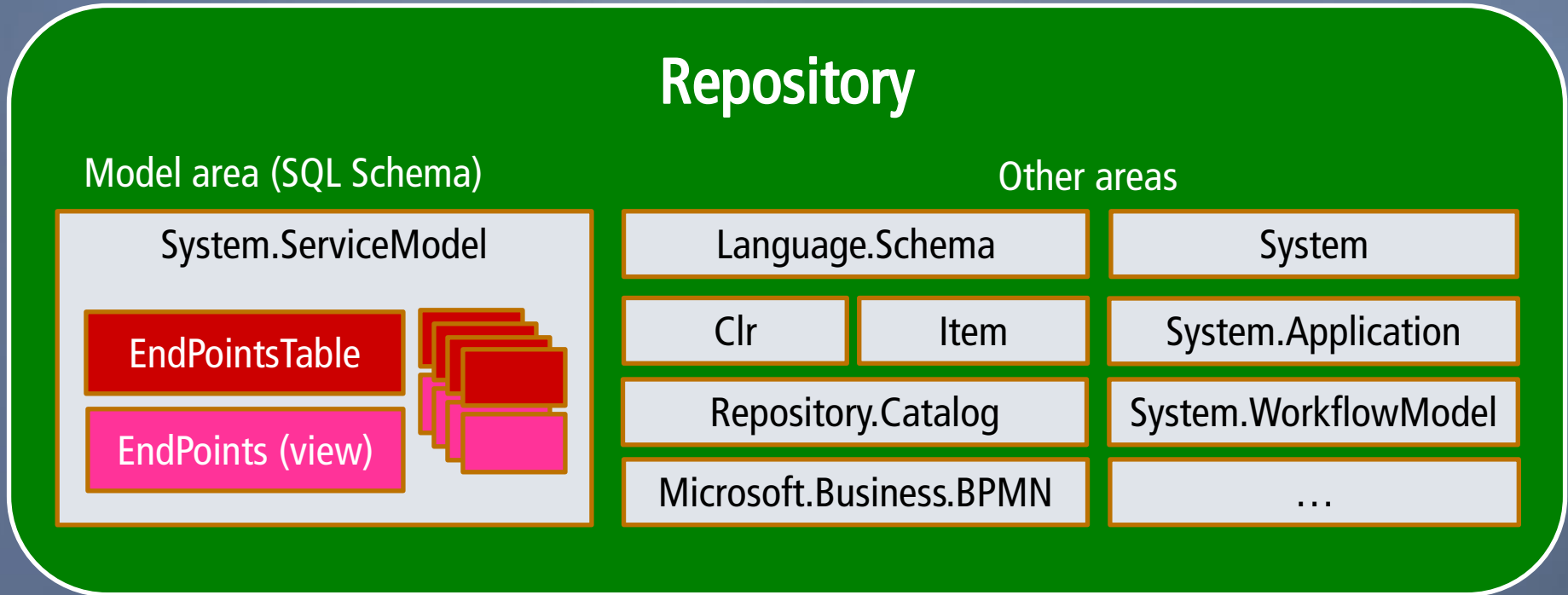
# Repository core services

- Deployment of models
  - Packages from M files in and out of repository
- Catalog of models
  - Extra information on modules, types and extents
  - Handy for deployment, enriched DAL generation
- Security for models and instances
  - Claims-based authentication and authorization
  - Secure views, auditing
- Support for versioning

# Other repository features

- Claims-based security
  - Various security roles: RepositoryOwner, -ReaderWriter, -User, ...
  - Secure views over tables
  - Auditing
- Globalization and localization support

# Structure of Oslo repository



- Separate schema for each problem domain
- Design patterns for tables, views and "instead-of" triggers
- Relationship between schemas by naming convention

# Working with models in repository

- Oslo "compilers" or API store model schemas and instances
- Model data inside repository is just data
- Use any data access method to read models
  - Raw ADO.NET
  - Object mappers

From M to M via M

# THE LANGUAGES

# "M": a family of modeling languages

- "M" broad set of capabilities to describe, validate, transform, access, and store data
- Three languages

**MSchema**

Type system

Defining  
model schemas

**MGraph**

Data model

Creating  
model instances

**MGrammar**

Transformation

Authoring domain  
specific languages

- Open Packaging Conventions (OSP)  
announcement of "M" specification

# MSchema

We do  
data,  
not  
metadata



Don Box

- Allows you to define schema of domain data models and queries over structured data
  - Values, constraints, and views
  - Natural projection to SQL (and easier)
- Structural type system for values
  - Not object oriented, no behavior (inheritance)
  - Not a new data access technology stack

# MGraph

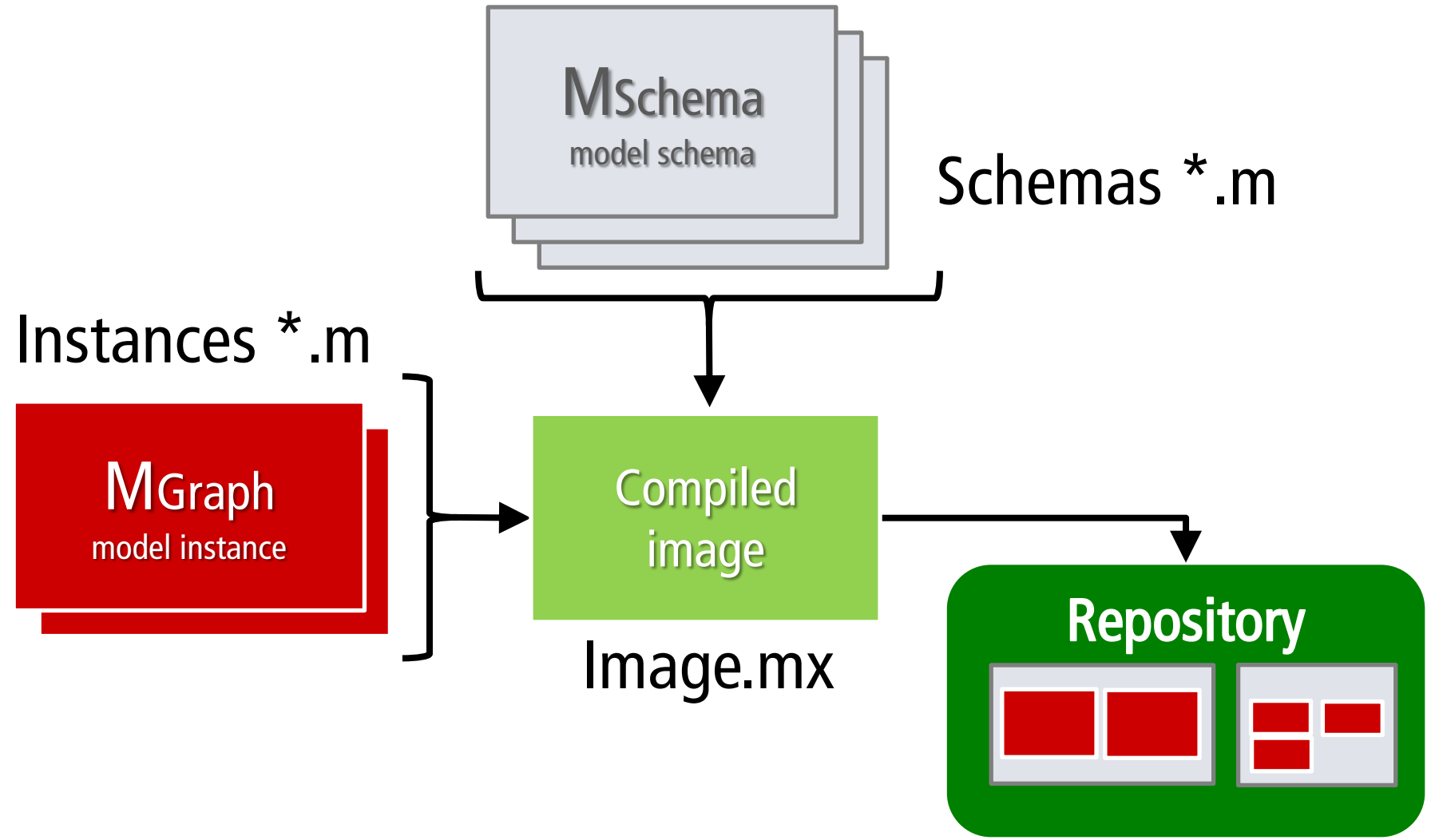
- Structural representation for values
  - Used with extent initializers and AST output of MGrammar transformation
- Labeled directed graph
- Familiar format: like JSON and object initializers

```

ServiceEndPoint {
  Address = "http://...",
  Binding {
    CloseTimeout = 10, ...
  }
}

```

# Language files and images



## M<sub>Grammar</sub>

- Everyone can build a parser for textual DSLs
- M<sub>Grammar</sub> defines projections from text to structured data
  - Rule-based transformation
  - Grammar-driven text editor integration
- Translation from DSL to M or XAML
- Composition (nesting) of languages
- Remember: DSLs may look like code, but it's just data

# MWorkflow DSL example

```

Activity PromptExact(PromptText: string, DesiredInput : string)
{
    input: string;
    inputMatched : bool;

    while (!inputMatched)
    {
        Prompt { Text = PromptText, Result = &input }
        inputMatched = input == DesiredInput;
    }
}

```

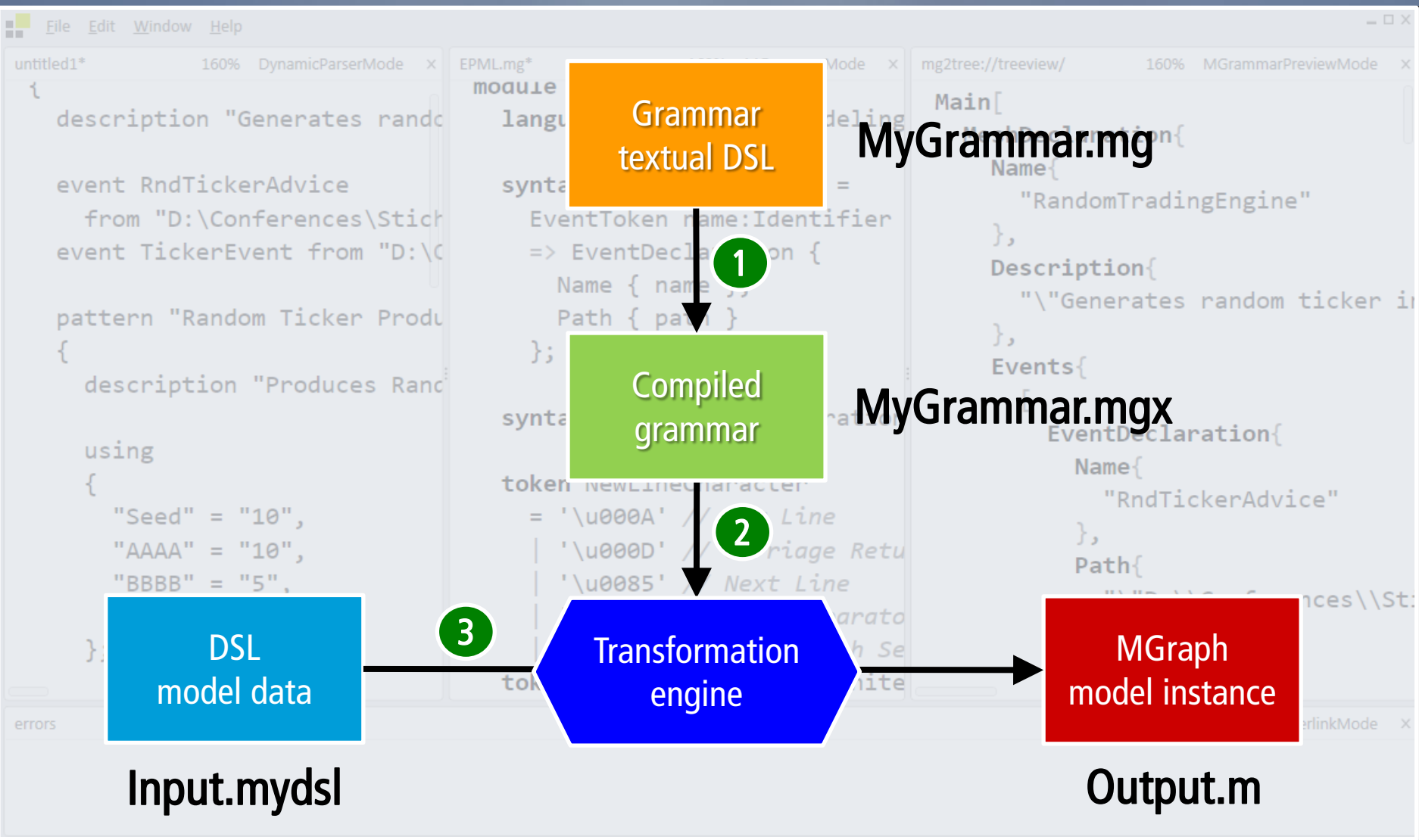
# MService example

```
module ClassA.OsloDemos
{
  import System;
  import System.ServiceModel;
  import System.ServiceModel.web;
  import System.Xml;

  service MyFirstOsloService
  {
    operation Echo(message : Text) : Text
    {
      WriteLine { Text = "Incoming message: "
        + message }
      return message;
    }
  }
}
```

MWorkflow?

# Working with DSLs



# Demo: an introduction to all languages

- Mschema and MGraph
  - Structural type system
  - Extents and instances
- MGrammar
  - Define a new Domain Specific Language
- Plus ... an sneak preview of the tooling

# THE TOOLS

# Command-line tools



- **m.exe: image compiler**
  - Compiles \*.m files into image or SQL



- **mx.exe: image loader**
  - Loads images into Oslo repository



- **mg.exe: grammar compiler**
  - Compiles grammar for DSL into image

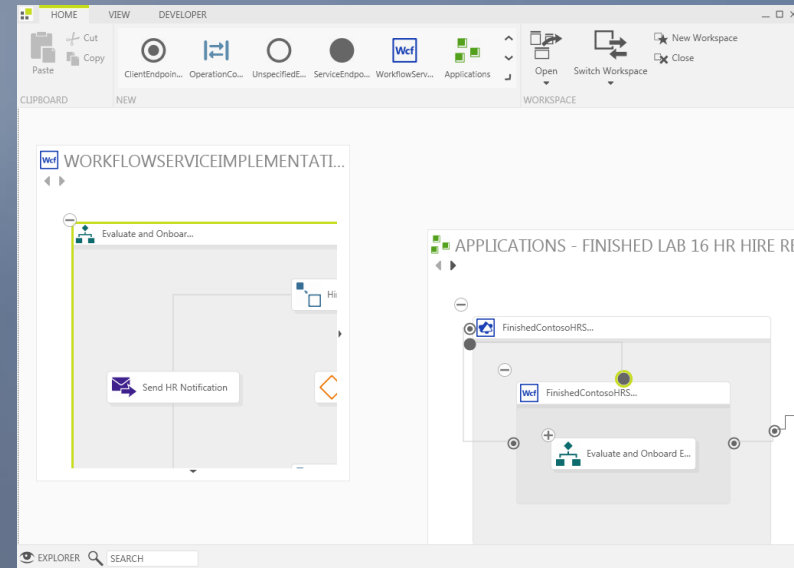


- **mgx.exe: grammar driver**
  - Transform DSL into MGraph or XAML instances



# Quadrant

- Tool to query, update and visualize data
  - *Flexible, focused design surfaces*
  - *Default experiences over arbitrary data*
- Quadrant is a model-driven application itself
  - Uses Oslo repository to store state, configuration and customizations



# Visual Studio

- Visual Studio enhancements
  - "M" Project and item template
  - New MSBuild targets for MGrammar and MSchema+MGraph
  - IntelliSense in M files
- Managed API for Oslo
  - Parsing and transformation engine
  - Building, reading MGraph instances

# Demo: playing around with models

- Quadrant
  - Editing models
  - Defining a composite application
  - Deployment
- Visual Studio
  - New build targets

# Review

- Oslo is the new modeling platform
- Express more of your app in data
- Languages to define and create models
- Tools to help create and editing models
- A repository to store and share models

## Learn more: additional resources

- Oslo Dev Center and Forums
- Team blogs: IntelliPad, Modeling language
- Team members
  - Don, Pinky, SpankyJ, SuperNinja, ChrisAn, Doug
- Samples:
  - Loads of samples in Oslo SDK
  - Spork      — Ultra flexible event pattern mesh
- Channel 9 Oslo videos (PDC and others)
- Class-A 828 event on Azure and Oslo

# Questions? Answers! Discussion!!!

?

!



# DSL Toolkit

- Version 4.0 ships with VS2010
  - File-based artifacts (schemas, model instances)
  - Graphical and forms-based designers as add-ins of VS
  - Shapes and shape mapping language
  - Output of graphical designers translated into code-based artifacts
  - Has its own lifecycle