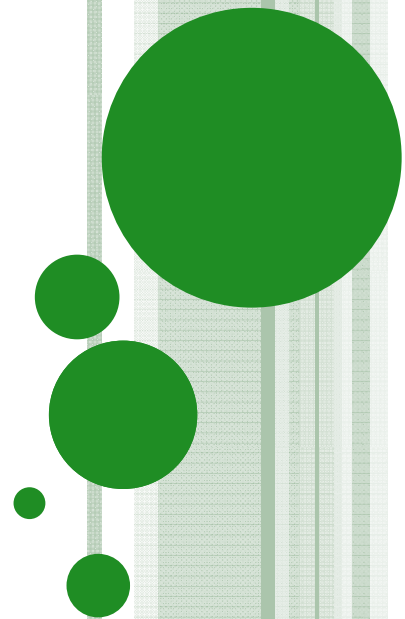


بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

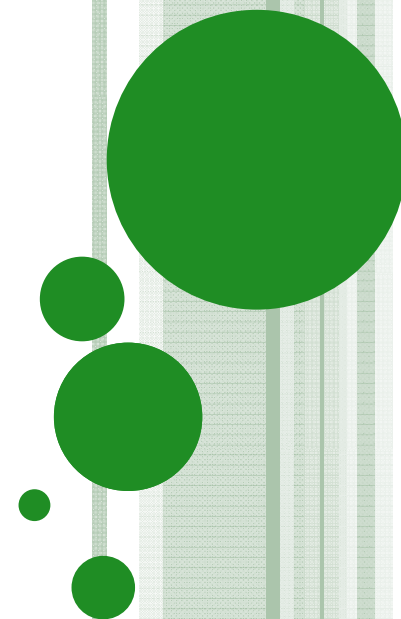


مفاهیم و ابزارها در معماری مدل محور

ارائه دهنده

علی جلال

همایش رقابت و فراگیری جاوا (JCAL) - آبان ۸۸



○ معماری مدل محور

○ Eclipse Modeling Framework(EMF)

• تعریف یک فرامدل

○ Open Architecture Ware(OAW)

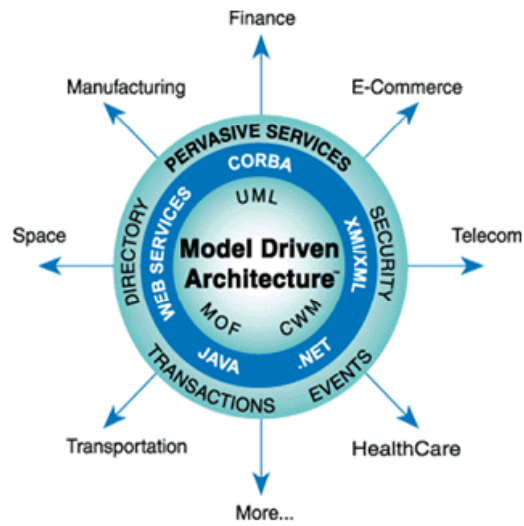
• Xpand Language

• Xtend Language

• Check Language

• Xtext Framework

• Recipe Framework



معماری مدل محور

Model Driven Architecture (MDA)

○ هر ساله تکنولوژی های جدیدی ابداع و به سرعت رایج می شوند:

- وابستگی یک سازمان به یک تکنولوژی خاص مشکل زا خواهد شد.
- برخی از نرم افزارها باید به تکنولوژیهای جدیدتر ارتقا یابند.

○ برخی سازمانها باید با تکنولوژی های مختلف در یک زمان کار کنند.

○ حفظ معماری یکسان در پروژه های مختلف یک سازمان ضروری است.

- یکسان سازی مدل و پیاده سازی نرم افزارهای مختلف یک سازمان مهم است.

○ طبق آمار CHAOS در سال ۲۰۰۹:

- ۴۴ درصد پروژه ها به مشکلاتی نظیر تأخیر، کسر بودجه و عملکرد ناصحیح نرم افزار بر می خورند.
- ۲۴ درصد پروژه ها لغو و یا هرگز مورد استفاده قرار نمی گیرند.
- تنها ۳۲ درصد پروژه ها به موقع، با بودجه تعیین شده و با عملکرد صحیح انجام می شوند.

○ شرکت های بزرگ غیر نرم افزاری از «خط تولید» برای تولیدات انبوه خود استفاده می کنند تا پاسخگوی تغییرات مداوم نیازها و تکنولوژیها باشند.

معماری مدل محور (MDA) چیست؟



○ تحقیقات مربوط به MDA در سال ۲۰۰۱ در OMG آغاز شد.

○ استفاده از زبان مدل سازی به عنوان زبان برنامه نویسی توصیفی

• افزایش سطح انتزاع برنامه نویسی

○ تولید کد بر پایه معماری مدل محور

• تبدیل مدل به:

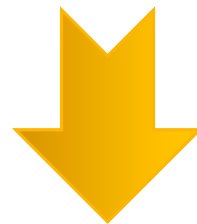
○ مدل

○ کد

Platform Independent Model (PIM)



Platform Specific Model (PSM)



کد قابل اجرا

مزیت های معماری مدل محور

○ افزایش قابلیت جابجایی بین سکوهای مختلف تولید نرم افزار

• مطابقت سریع با تکنولوژی های نوظهور

○ کاهش هزینه ها و زمان اجرای پروژه

• کاهش خطاهای احتمالی که با تولید دستی کدها بوجود می آیند.

○ سازگاری و یکنواختی بالا در کدهای تولید شده و ایجاد معماری یکسان

• استانداردسازی خودکار برای نرم افزارهای تولیدی یک شرکت

مزیت های معماری مدل محور (ادامه)

○ جداسازی محدوده مسئله و محدوده پاسخ در تولید نرم افزار

- تمرکز بیشتر بر روی مدلسازی و بهبود طراحی نرم افزار

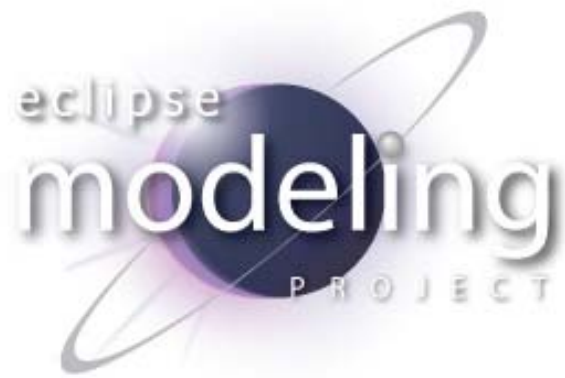
○ همگام سازی مدل های تحلیل و یا طراحی با کدهای تولید شده

- امکان مهندسی مجدد نرم افزار برای تولید نرم افزار بر پایه تکنولوژی های جدید

○ در سیستم های اطلاعاتی حدود ۵۰ درصد نیازمندی ها مربوط به عملیات

CRUD بر روی موجودیتهای سیستم است و می توان با کمک تولید کننده

کد آنها بر آورده نمود.

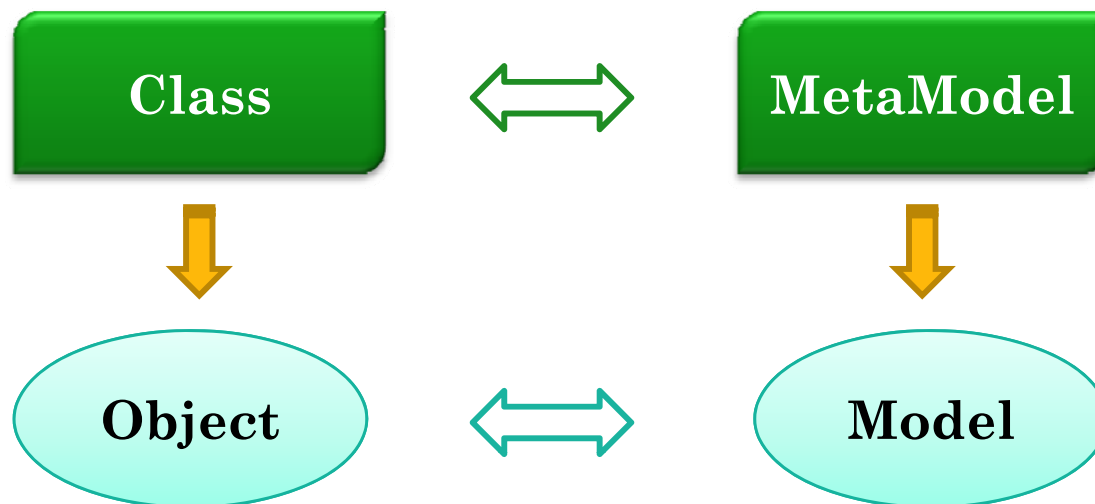


EMF

Eclipse Modeling Framework

○ EMP (پروژه مدلسازی در محیط برنامه سازی Eclipse) بر روی توسعه و تولید نرم افزارها بر پایه مدل تمرکز دارد.

○ EMF یک چارچوب مدلسازی و تولید کد برای ساختن ابزارها و برنامه های مبتنی بر مدل است.



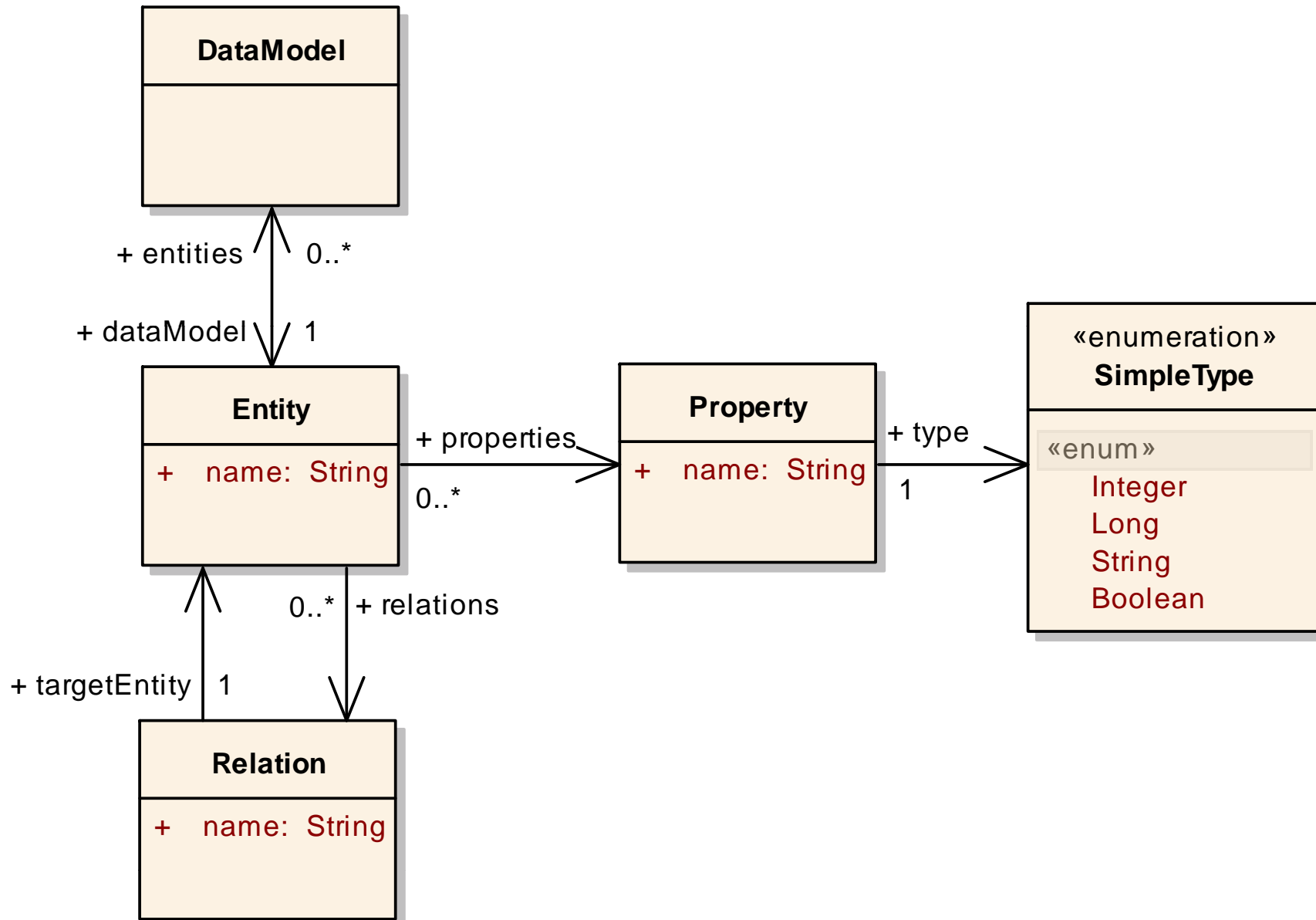
○ رابطه مدل و فرامدل:

○ Ecore یک چارچوب تعریف فرامدل در EMF است.

○ با تعریف یک فرامدل دلخواه، EMF ویرایشگر دیداری مدل های مربوط به فرامدل را در اختیار ما می گذارد.

○ مزیت فرامدل های Ecore نسبت به Class Diagram:

- تعریف فرامدل دلخواه
- تولید خودکار ویرایشگر مدل مورد نظر





OAW

Open Architecture Ware

○ **OAW** مجموعه ای از ابزارها و تکنولوژیها می باشد که هدف اصلی آن توسعه و تولید نرم افزارهای مدل محور (و تولید کد) می باشد.

○ **OAW** همانند **EMF** بخشی از پروژه مدلسازی ابزار برنامه نویسی **Eclipse** می باشد.

○ **OAW** با مدل‌های **EMF** سازگار است و همه ابزارها و ویرایشگرهای آن بر مبنای محیط **Eclipse** تعریف شده اند.

- خواندن فرامدل‌های مختلف:
 - Ecore، UML2، JavaClass، XSD و ...
- تعریف قالب برای تولید کد
- تبدیل مدل به کد
- تبدیل مدل به مدل دیگر
- اعتبارسنجی و افزودن محدودیت در زمان تولید کد و بعد از آن
- معرفی یک تولید کننده Parser برای زبانهای دلخواه
- امکانات برنامه نویسی جنبه گرا در حین تولید کد

- برای اجرای موتور تولید کد در OAW به تعریف Workflow نیاز داریم.
- یک فایل با فرمت XML، مولفه های مورد استفاده در تولید کد و ترتیب اجرای آنها را توصیف می کند.
- این فایل نقطه ورود برای تولید کد می باشد و مولفه های پردازش بخشهای مختلف در این فایل تعریف می شود.
- مانند کلاسهای خواندن فرامدل، پردازش قالب، چک کننده مدل و کد خروجی، تولید کننده کد و ...

```
<workflow>
  <component id="xmiParser"
    class="org.openarchitectureware.emf.XmiReader">
    <modelFile value="example.model"/>
    ...
  </component>
  ...
  <component
    class="org.openarchitectureware.xpand2.Generator">
    <metaModel class="...type.emf.EmfMetaModel"/>
    ...
    <expand value="Templates::Root FOR model"/>
      <outlet path="src-gen"/>
        <postprocessor
          class="...xpand2.output.JavaBeautifier"/>
        </outlet>
      <expand/>
    </component>
  </workflow>
```

- Xpand یک زبان تعریف قالب برای تولید کد خروجی است.
- ویرایشگر آن نسبت به فرامدل حساس است و Code Completion و Syntax Highlighting را در اختیار می گذارد.
- فراهم کردن امکان Debug
- امکان صدا زدن قالبهای دیگر در یک قالب
- امکان تعریف Protected Region برای جلوگیری از Overwrite


```
«IMPORT mypackage::metamodel»  
«EXTENSION mypackage::ExtensionFile»  
  
«DEFINE javaClass FOR Entity»  
  «FILE getFileName()»  
    package «getJavaPackage()» ;  
    public class «name» {  
      // some implementation  
    }  
  «ENDFILE»  
«ENDDEFINE»
```

- Xtend یک زبان تعریف متدهایی برای استفاده در قالبها است تا از پیچیدگی قالبها و فرامدل جلوگیری کند.
- یک متد Xtend می تواند از متدهای سایر فایل‌های Xtend، متدهای جاوا یا متدهای بازگشتی استفاده کند.
- ویرایشگر آن نسبت به فرامدل حساس است و Code Completion و Syntax Highlighting را در اختیار می گذارد.
- امکان Debug در اختیار ما می گذارد.

```
import mypackage::metamodel;  
extension mypackage::ExtensionFile;  
  
/**  
 * Documentation  
 */  
anExpressionExtension(String stringParam) :  
    doSomethingWith(stringParam);  
  
String aJavaExtension(String param) :  
    JAVA my.JavaClass.staticMethod(param);
```

○ **Check** زبان تعریف تعریف محدودیت بر روی مدل ورودی است.

○ دو نوع خطا تعریف می کند: **Error** و **Warning**

○ ویرایشگر آن نسبت به فرامدل حساس است و **Code Completion** و

Syntax Highlighting را در اختیار می گذارد.

○ امکان **Debug** در اختیار ما می گذارد.

○ امکان استفاده از متدهای **Xtend** در آن وجود دارد.

```
import mypackage::metamodel;  
extension mypackage::ExtensionFile;
```

```
context Attribute ERROR  
  "Names have to be more than one  
  character long." :  
  name.length > 1;
```

○ Xtext به ما امکان تعریف زبانهای مختص یک حوزه (DSL) را می دهد و خروجی های زیر را تولید می کند:

- Parser مختص زبان که به طور مستقل قابل استفاده است.
- ویرایشگر متنی مختص زبان
- فایل Ecore متناظر با تعریف زبان و در نتیجه امکانات تولید کد بر مبنای زبان مورد نظر

○ ویرایشگر آن امکاناتی نظیر **Code Completion**، **Outline**، **Navigation** مبتنی بر زبان مورد نظر را ارائه می دهد.

برنامه نویسی جنبه گرا در OAW

○ در برنامه نویسی جنبه گرا می توانیم کارهای مشابه را با کمک تعریف جنبه ها انجام دهیم مانند Logging.

○ بدون تغییر مستقیم تولیدکننده کد، می توان به آن بخشهایی اضافه کرد.

○ قالب کلی تعریف جنبه در Xtend:

```
around my::somePrefix*() :  
    // a sequence of statements ->  
    ctx.proceed();
```

○ قالب کلی تعریف جنبه در Xpand:

```
«AROUND my::Template* FOR type»  
    // a sequence of statements  
«ENDAROUND»
```

- در حال حاضر نمی توان به کمک MDA، کل پروژه را پیاده سازی کرد.
- Recipe امکانی برای کنترل کلاسهای نوشته شده توسط برنامه نویس در محیط Eclipse است تا پس از فرایند تولید کد نیز بتوان محدودیت هایی را بر روی کلاسها تعریف کرد.
- در یک کلاس محدودیت های مورد نظر خود را تعریف می کنیم و پس از اجرای Workflow، نتیجه در فایل با پسوند recipes ذخیره می شود.
- فایل recipes بعد از هر ذخیره، ایجاد و یا حذف فایل در محیط Eclipse یک بار به روز رسانی می شود.
- پیام های خطا و یا موفقیت شرایط در پنجره مخصوصی نمایش داده می شود.

- www.omg.org/mda
- www.eclipse.org/modeling/emf
- www.openarchitectureware.org
- www.eclipse.org/workinggroups/oaw