

# Meta Programming

Nelson Ferraz  
nferraz@gmail.com

# Resumo

- Meta Programming
- Code Generation
- AppML

# Meta Programming

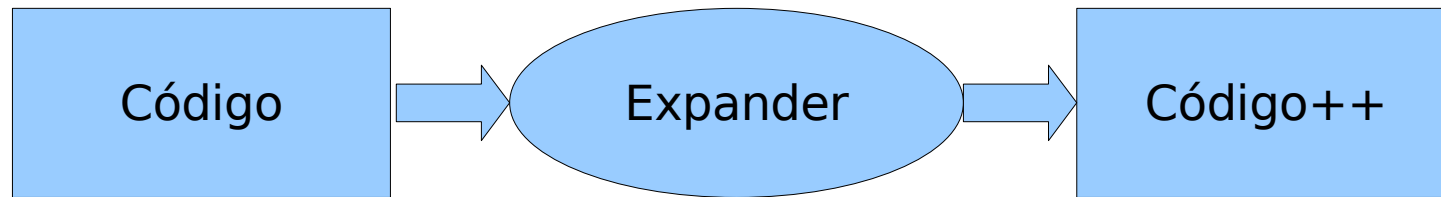
- Programas que manipulam programas
  - Geração de código
  - Em tempo de compilação
  - Ou mesmo em runtime

# Vantagens

- Qualidade
- Consistência
- Produtividade
- Abstração

# Inline Code Expander

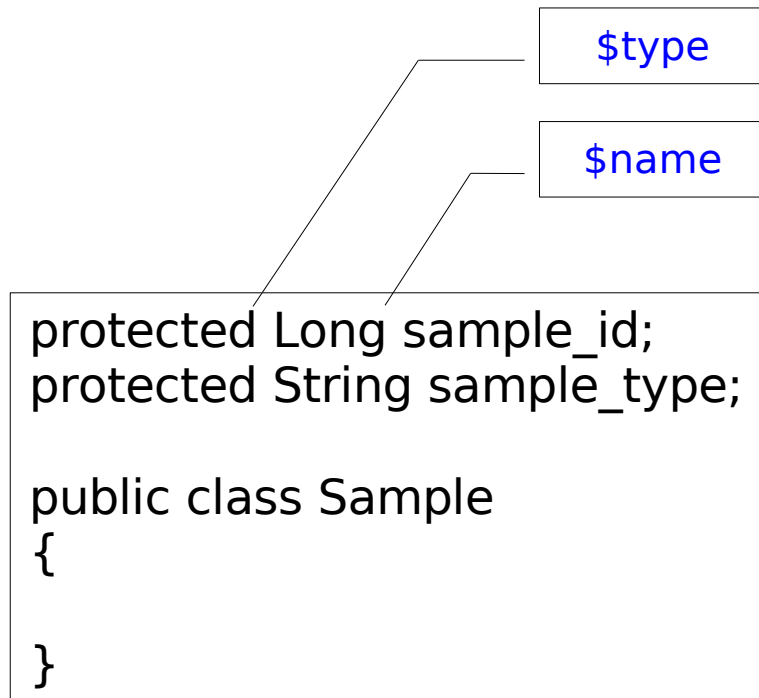
# Inline Code Expander



# Inline Code Expander

```
protected Long sample_id;  
protected String sample_type;  
  
public class Sample  
{  
  
}
```

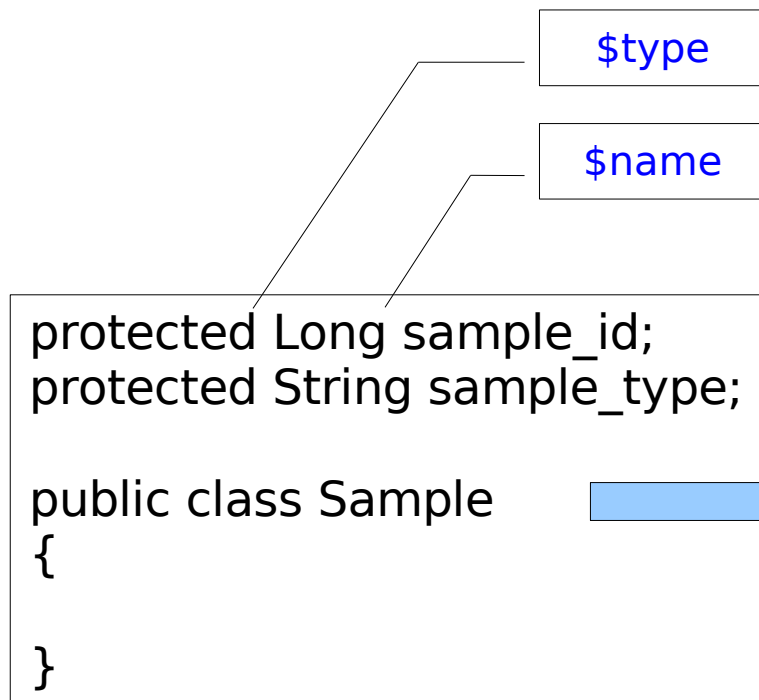
# Inline Code Expander





# Inline Code Expander

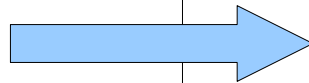
```
protected Long sample_id;  
protected String sample_type;  
  
public class Sample  
{  
  
}
```



```
public class Sample  
{  
    public void set_$name($type p_$name)  
    {  
        $name = p_$name;  
    }  
  
    public $type get_$name()  
    {  
        return $name;  
    }  
}
```

# Inline Code Expander

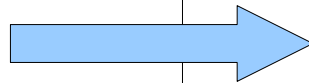
```
protected Long sample_id;  
protected String sample_type;  
  
public class Sample  
{  
  
}
```



```
public class Sample  
{  
    public void setSample_id(Long p_sample_id)  
    {  
        sample_id = p_sample_id;  
    }  
  
    public Long getSample_id()  
    {  
        return sample_id;  
    }  
}
```

# Inline Code Expander

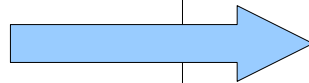
```
protected Long sample_id;  
protected String sample_type;  
  
public class Sample  
{  
  
}
```



```
public class Sample  
{  
    public void setSample_id(Long p_sample_id)  
    {  
        sample_id = p_sample_id;  
    }  
  
    public Long getSample_id()  
    {  
        return sample_id;  
    }  
  
    public void setSample_type(String  
p_sample_type)  
    {  
        sample_type = p_sample_type;  
    }  
  
    public String getSample_type()  
    {  
        return sample_type;  
    }  
}
```

# Inline Code Expander

```
protected Long sample_id;  
protected String sample_type;  
  
public class Sample  
{  
  
}
```



```
protected Long sample_id;  
protected String sample_type;  
  
public class Sample  
{  
    public void setSample_id(Long p_sample_id)  
    {  
        sample_id = p_sample_id;  
    }  
  
    public Long getSample_id()  
    {  
        return sample_id;  
    }  
  
    public void setSample_type(String  
p_sample_type)  
    {  
        sample_type = p_sample_type;  
    }  
  
    public String getSample_type()  
    {  
        return sample_type;  
    }  
}
```

# Code Transformation



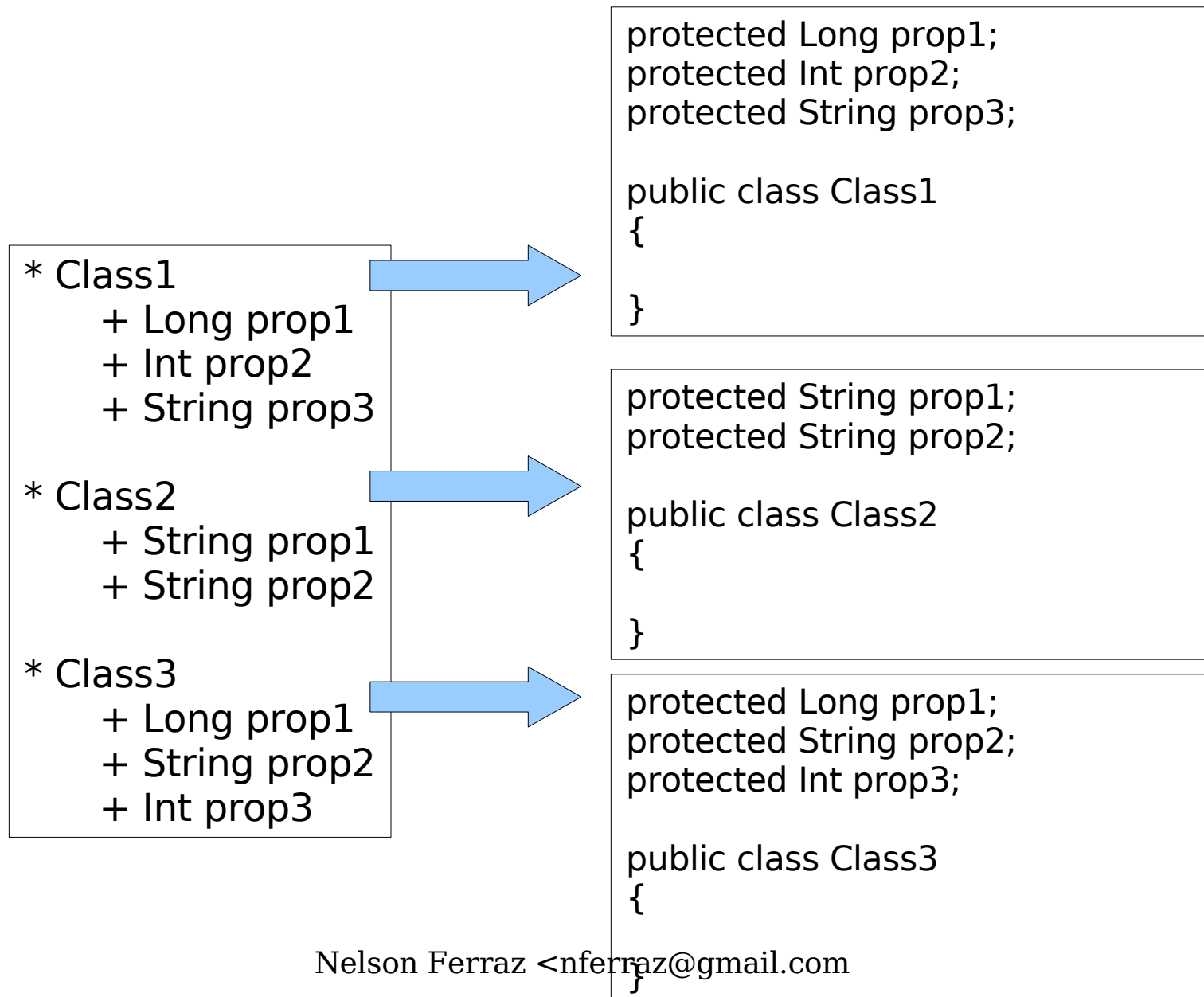
# Code Transformation

```
* Class1
  + Long prop1
  + Int prop2
  + String prop3

* Class2
  + String prop1
  + String prop2

* Class3
  + Long prop1
  + String prop2
  + Int prop3
```

# Code Transformation

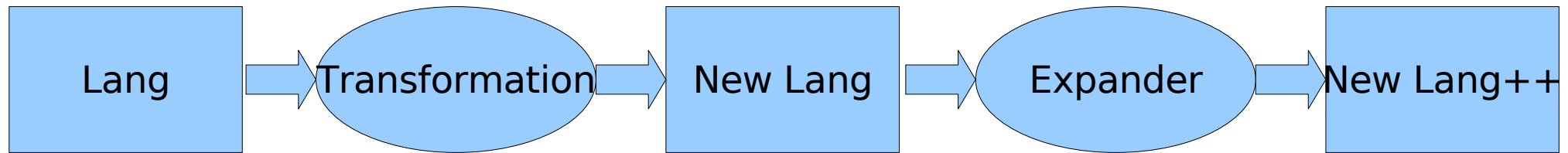


# “Filosofia Unix”

- Ferramentas pequenas
- Fazem apenas uma coisa (e bem)
- Funcionam como filtros: lêem textos e escrevem textos



# “Filosofia Unix”



```
* Class1
  + Long prop1
  + Int prop2
  + String prop3

* Class2
  + String prop1
  + String prop2

* Class3
  + Long prop1
  + String prop2
  + Int prop3
```

```
protected Long
prop1;
protected Int prop2;
protected String
prop3;

public class Class1
{
}
```

```
protected Long sample_id;
protected String sample_type;

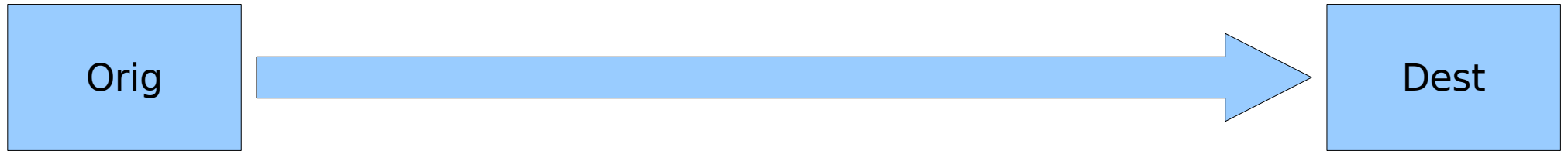
public class Class1
{
  public void setSample_id(Long p_sample_id)
  {
    sample_id = p_sample_id;
  }

  public Long getSample_id()
  {
    return sample_id;
  }

  public void setSample_type(String
p_sample_type)
  {
    sample_type = p_sample_type;
  }

  public String getSample_type()
  {
    return sample_type;
  }
}
```

# “Filosofia Unix”



```
* Class1
  + Long prop1
  + Int prop2
  + String prop3

* Class2
  + String prop1
  + String prop2

* Class3
  + Long prop1
  + String prop2
  + Int prop3
```

23/11/07

```
protected Long sample_id;
protected String sample_type;

public class Class1 {
{
  public void setSample_id(Long p_sample_id)
  {
    sample_id = p_sample_id;
  }

  public Long getSample_id()
  {
    return sample_id;
  }

  public void setSample_type(String
p_sample_type)
  {
    sample_type = p_sample_type;
  }

  public String getSample_type()
  {
    return sample_type;
  }
}
```

Orig



Dest

```
* Class1
  + Long prop1
  + Int prop2
  + String prop3

* Class2
  + String prop1
  + String prop2

* Class3
  + Long prop1
  + String prop2
  + Int prop3
```

```
CREATE TABLE Class1 (
  prop1 type1,
  prop2 type2,
  prop3 type3,
);

CREATE TABLE Class1 (
  prop1 type1,
  prop2 type2,
);

CREATE TABLE Class1 (
  prop1 type1,
  prop2 type2,
  prop3 type3,
);
```

Orig



Dest

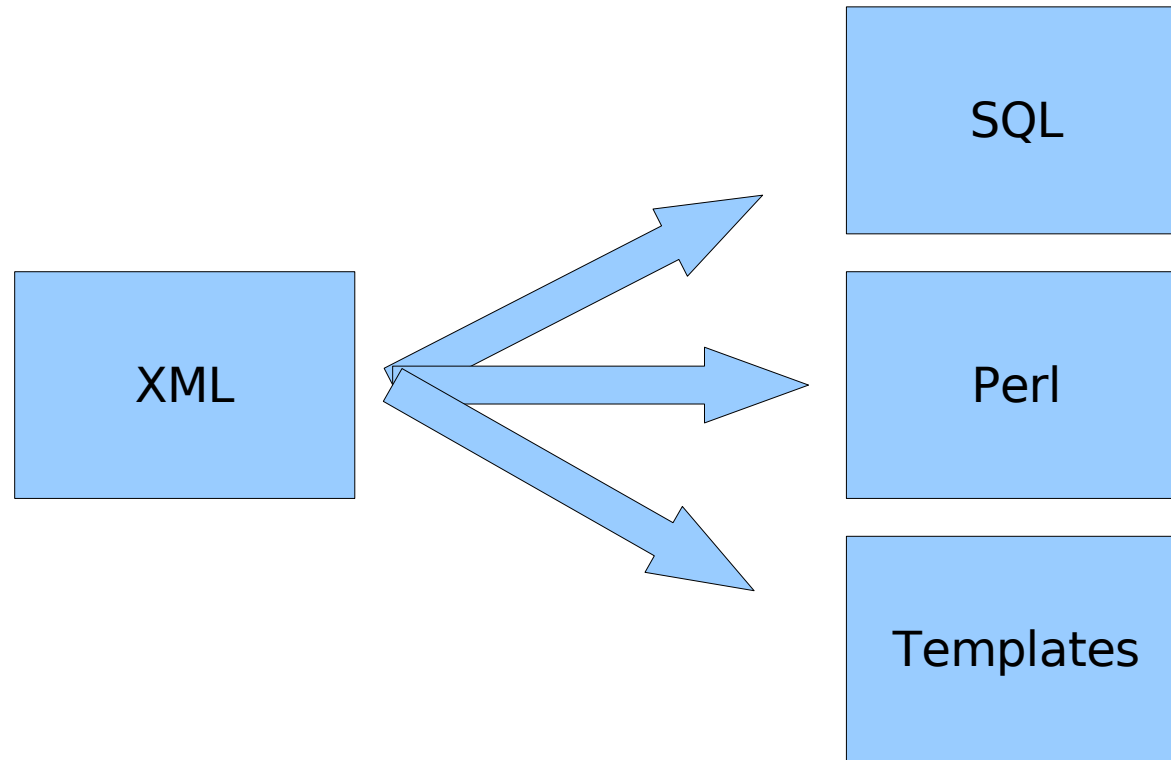
- \* **Class1**
  - + Long prop1
  - + Int prop2
  - + String prop3
- \* **Class2**
  - + String prop1
  - + String prop2
- \* **Class3**
  - + Long prop1
  - + String prop2
  - + Int prop3

23/11/07

```
class1.html  
<h1>Class1</h1>  
  
<form>  
  prop1: <input name="prop1"  
type="text" size="40"/>  
  prop2: <input name="prop2"  
type="text" size="40"/>  
  prop3: <input name="prop3"  
type="text" size="40"/>  
</form>  
  
class2.html  
<h1>Class2</h1>  
  
<form>  
  prop1: <input name="prop1"  
type="text" size="40"/>  
  prop2: <input name="prop2"  
type="text" size="40"/>  
  prop3: <input name="prop3"  
type="text" size="40"/>  
</form>
```

# AppML

- Application Markup Language
- Uma fonte, múltiplas saídas:



# AppML

- agenda.xml

```
<project name="agenda">
```

```
</project>
```

# AppML

- agenda.xml

```
<project name="agenda">  
  <table name="contato">  
  </table>  
</project>
```

# AppML

- agenda.xml

```
<project name="agenda">  
  <table name="contato">  
    <field name="nome"/>  
    <field name="endereco"/>  
    <field name="telefone"/>  
  </table>  
</project>
```



# AppML

- agenda.xml

```
<project name="agenda">  
  <table name="contato">  
    <field name="nome" type="varchar" size="40"/>  
    <field name="endereco" type="varchar"  
size="40"/>  
    <field name="telefone" type="varchar" size="40"/>  
  </table>  
</project>
```

# AppML

- agenda.xml

```
<project name="agenda">  
  <table name="contato">  
    <field name="nome" type="varchar" size="40"/>  
    <field name="endereco" type="varchar"  
size="40"/>  
  </table>  
  
  <table name="telefone">  
    <rel name="pessoa" table="contato"/>  
    <field name="telefone" type="varchar" size="40"/>  
  </table>  
  
</project>
```

# AppML

- create.sql

```
CREATE TABLE contato (  
  id serial primary key,  
  nome varchar(40),  
  endereco varchar(40),  
  created timestamp DEFAULT NOW(),  
  updated timestamp,  
  deleted timestamp  
);
```

```
CREATE TABLE telefone (  
  id serial primary key,  
  pessoa int references contato,  
  telefone varchar(40),  
  created timestamp DEFAULT NOW(),  
  updated timestamp,  
  deleted timestamp
```

# AppML

- view.sql

```
CREATE VIEW view_telefone AS
SELECT
  telefone.telefone,
  telefone.pessoa,
  pessoa.nome AS pessoa_nome,
  pessoa.endereco AS pessoa_endereco
FROM telefone
LEFT JOIN contato ON (contato.id = telefone.pessoa);
```

# AppML

- frm\_telefone.tt2

(...)

```
<form method="post" action="[% url %]/agenda/telefone/list_telefone"
onSubmit="return(Validation())">
<input type="hidden" name="action" value="[% todo_action %]">
<input type="hidden" name="table" value="telefone">
<input type="hidden" name="id" value="[% my_telefone.id() %]">

<table>

<tr>
<th class="frm"></th>
<td class="frm">
  <select name="pessoa">
    <option value="">_____</option>
[% FOREACH row = list_contato %]
    <option value="[% row.id() %]" [% "selected" IF my_telefone.pessoa() == row.id() %]> [% row.() %]
[% END %]
  </select>
[% IF (show_insert_contato == 1) %]
  <a href="[% url %]/agenda/contato/frm_contato?todo_action=create">insert</a>
[% END %]
</td>
</tr>
```

(...)

# AppML

- telefone.pm

```
package agenda::cdbi::telefone;

# Project:
# Module:
# Date:

use strict;
use base qw( agenda::cdbi );

__PACKAGE__->table('telefone');

__PACKAGE__->columns(All => qw(id pessoa telefone));

__PACKAGE__->has_a(pessoa => 'agenda::cdbi::pessoa');

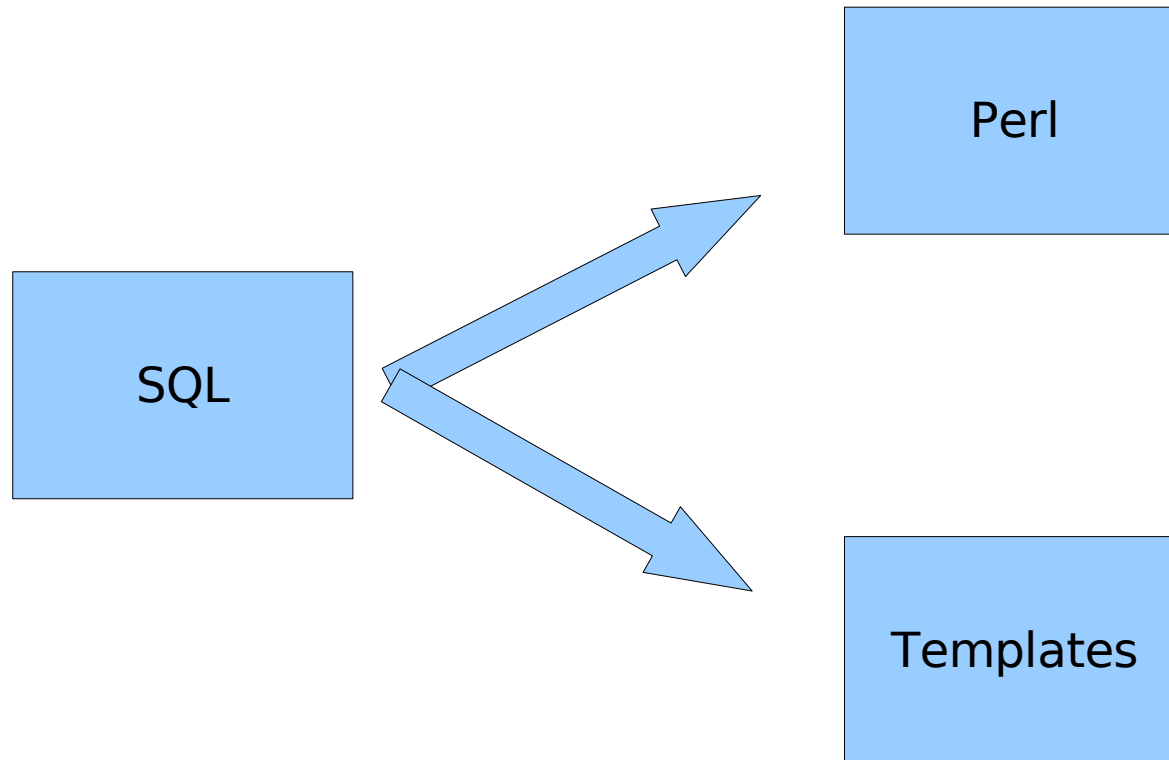
__PACKAGE__->sequence('telefone_id_seq');

1;
```

# SQL::Translator

- Extrai informações diretamente do SQL
- Pode ser extendido através de Producers

# SQL::Translator::Producer::App





# SQL::Translator::Producer::App