NSPYRE

making technology matter

SW: from Craftsmanship to Industry

The role of code generation in the industrialisation of software development

NSPYRE

making technology matter

The Market Situation

Complexity grows

Shorter TTM

15% - 25% of all software defects are delivered to customers.

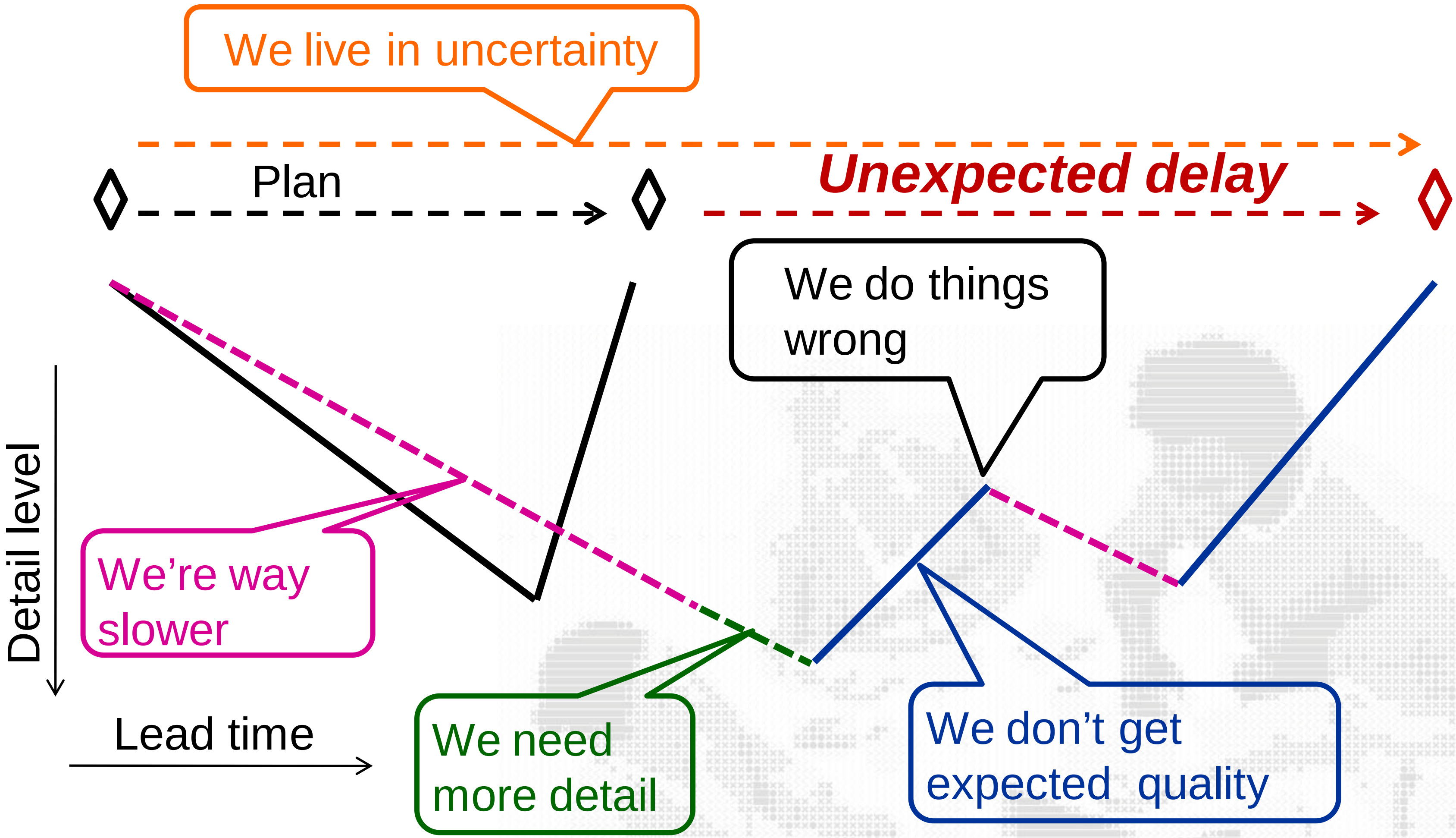
Approximately 40-50% of the effort is spent on avoidable work.

Labor-intensive, requiring highly educated engineers.

marc.hamilton@nspyre.nl

NSPYRE
making technology matter

The V-model: Current practice





NSPYRE

making technology matter

Software Productivity Killers

- Poor design: Software specifications and designs are not *verified* before implementation
  - Incomplete, incorrect and ambiguous specifications and designs are only discovered during the test phases
- High Complexity: Software designs are increasingly asynchronous, concurrent, reactive and event driven
  - Complexity, Deadlocks, Races, Non-determinism, Multi-core CPUs
  - Software architects can't oversee implications of design choices
- Poor Quality: Testing is an exercise in sampling
  - Sample is small, population is very large
  - A linear increase in the probability of finding a defect requires an exponential increase in the number of test cases
- Testing is the most expensive, least certain way to detect and remove defects and has the biggest impact on T2M

Software Engineering Challenge

Product specification — Highly variable, human language based dialects

“Design Gap”

Program code — Limited variable, general applicable artificial dialects

Level of Automation

(C)OTS compilers, interpreters

Machine code — Limited variable, processor-family dependent dialects

Reduce the “Design Gap”

Product specification

“Design Gap”

Product code

Specific artificial dialect

Specific compilers, interpreters

Program code

Level of Automation

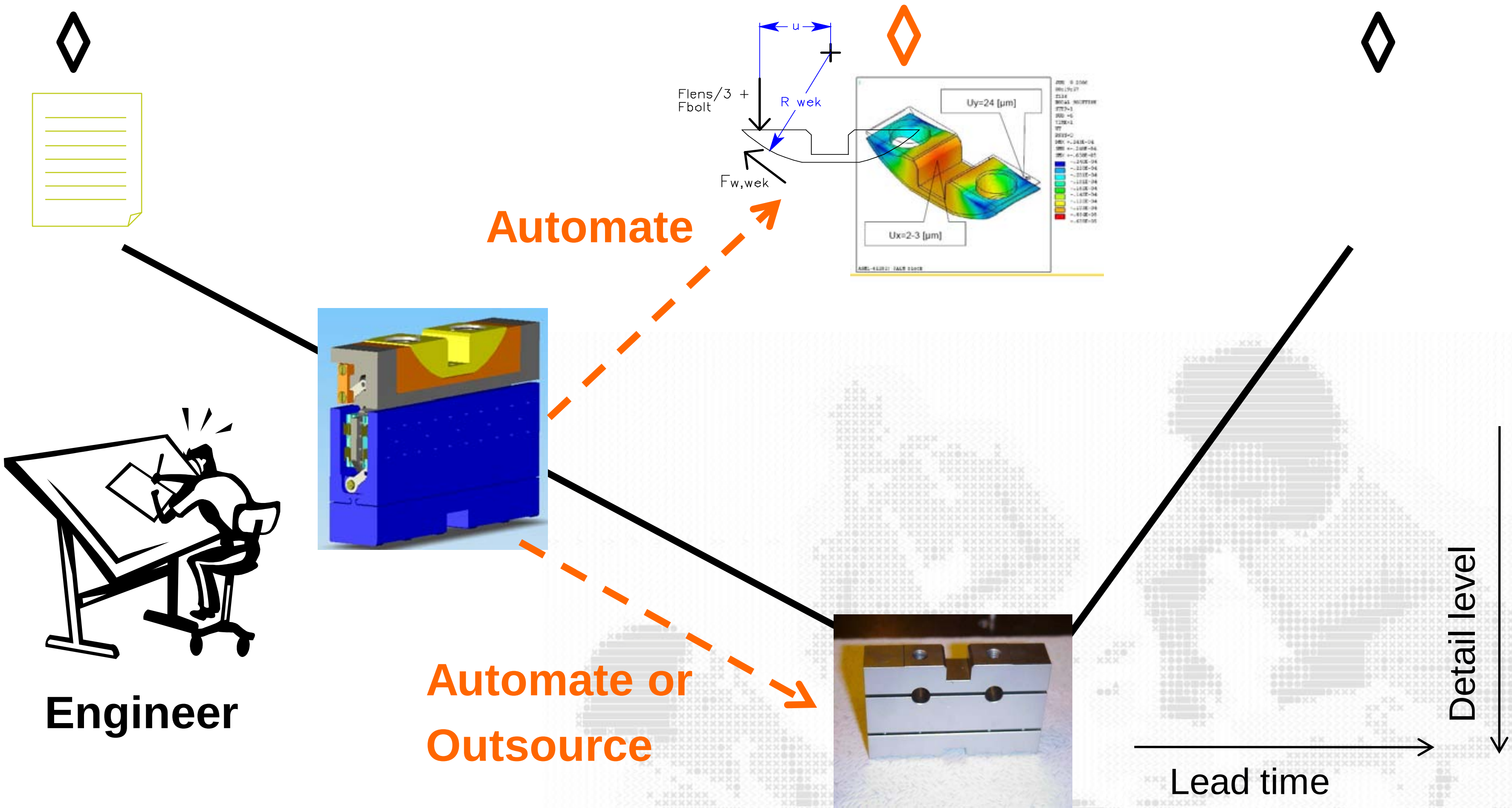
Machine code

marc.hamilton@nspyre.nl

NSPYRE

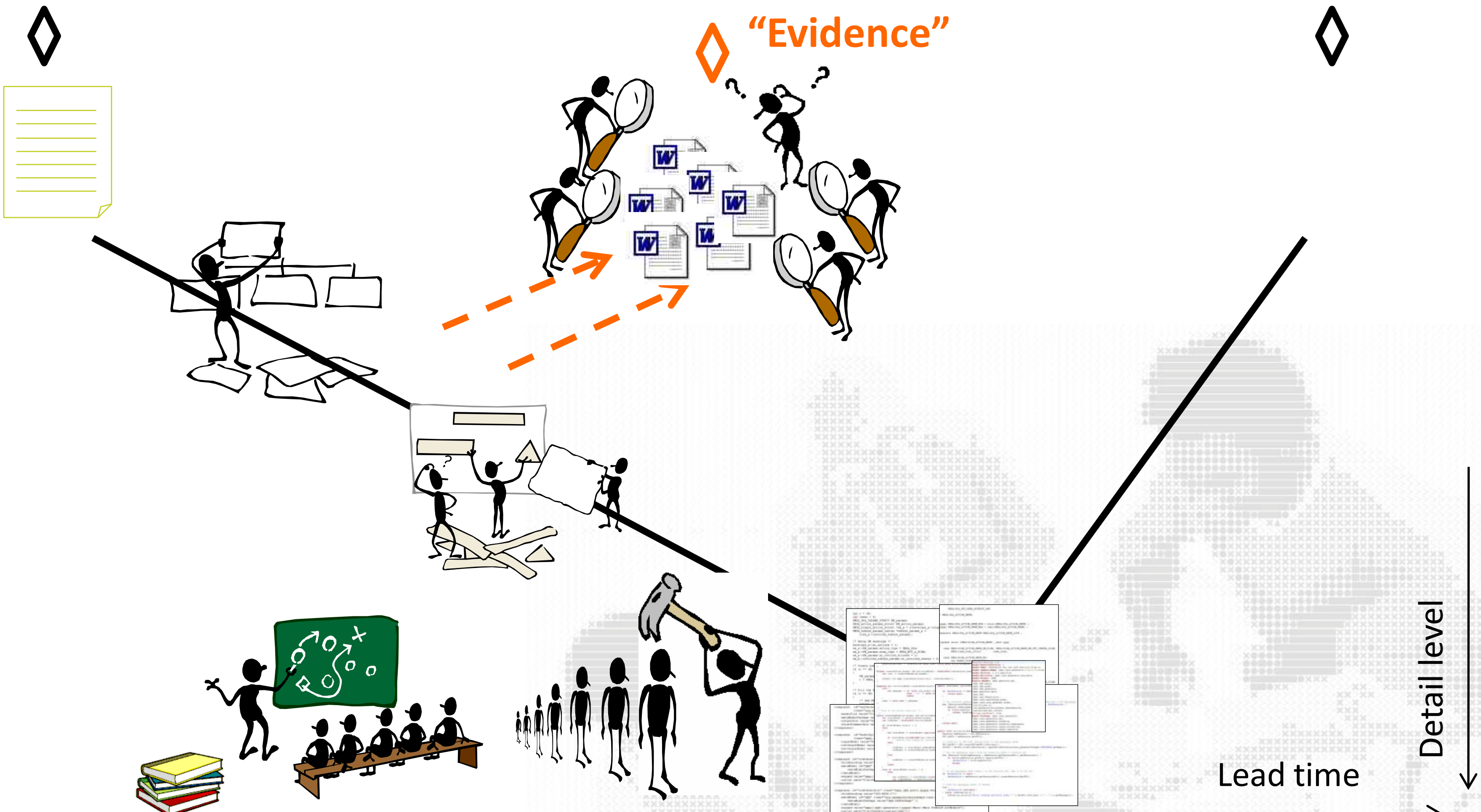
making technology matter

This is Mechanical Engineering



marc.hamilton@nspyre.nl

This is Software Engineering

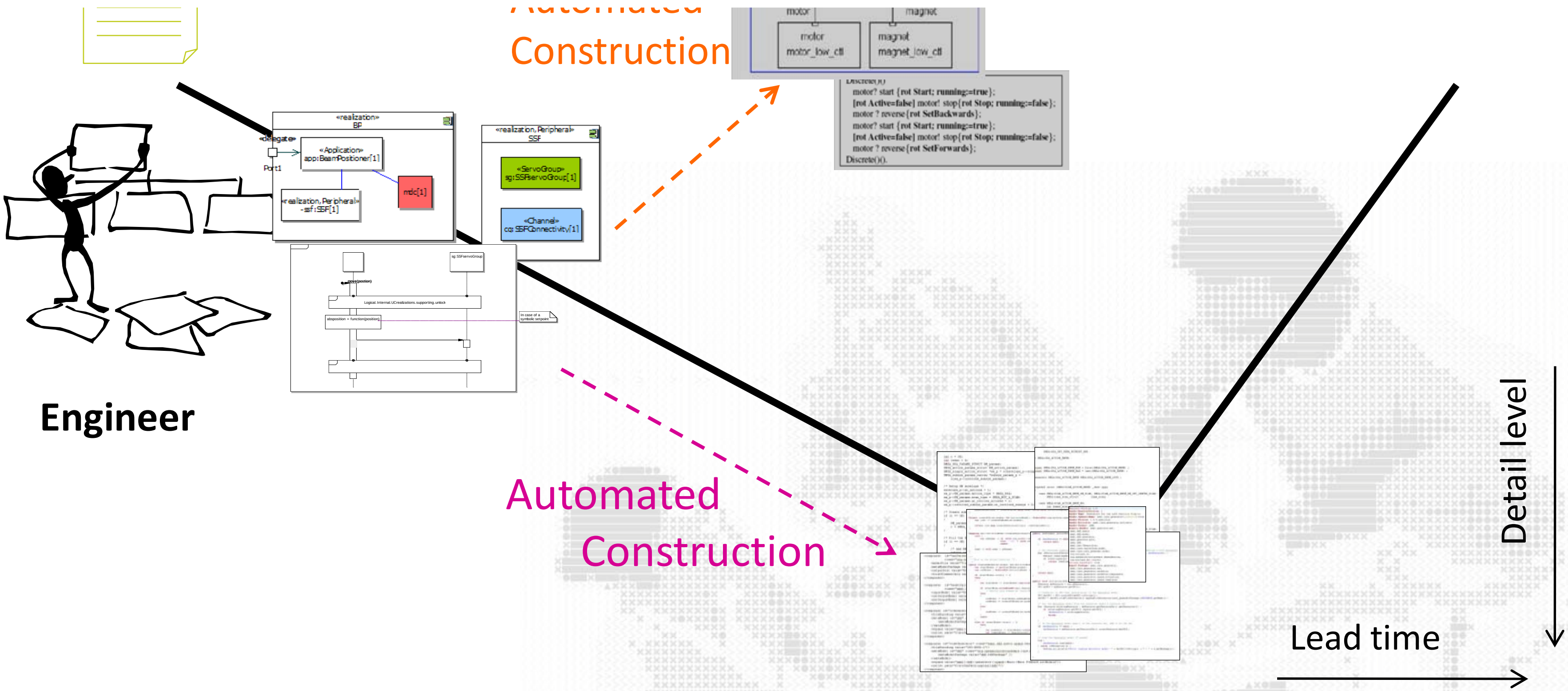


NSPYRE

making technology matter

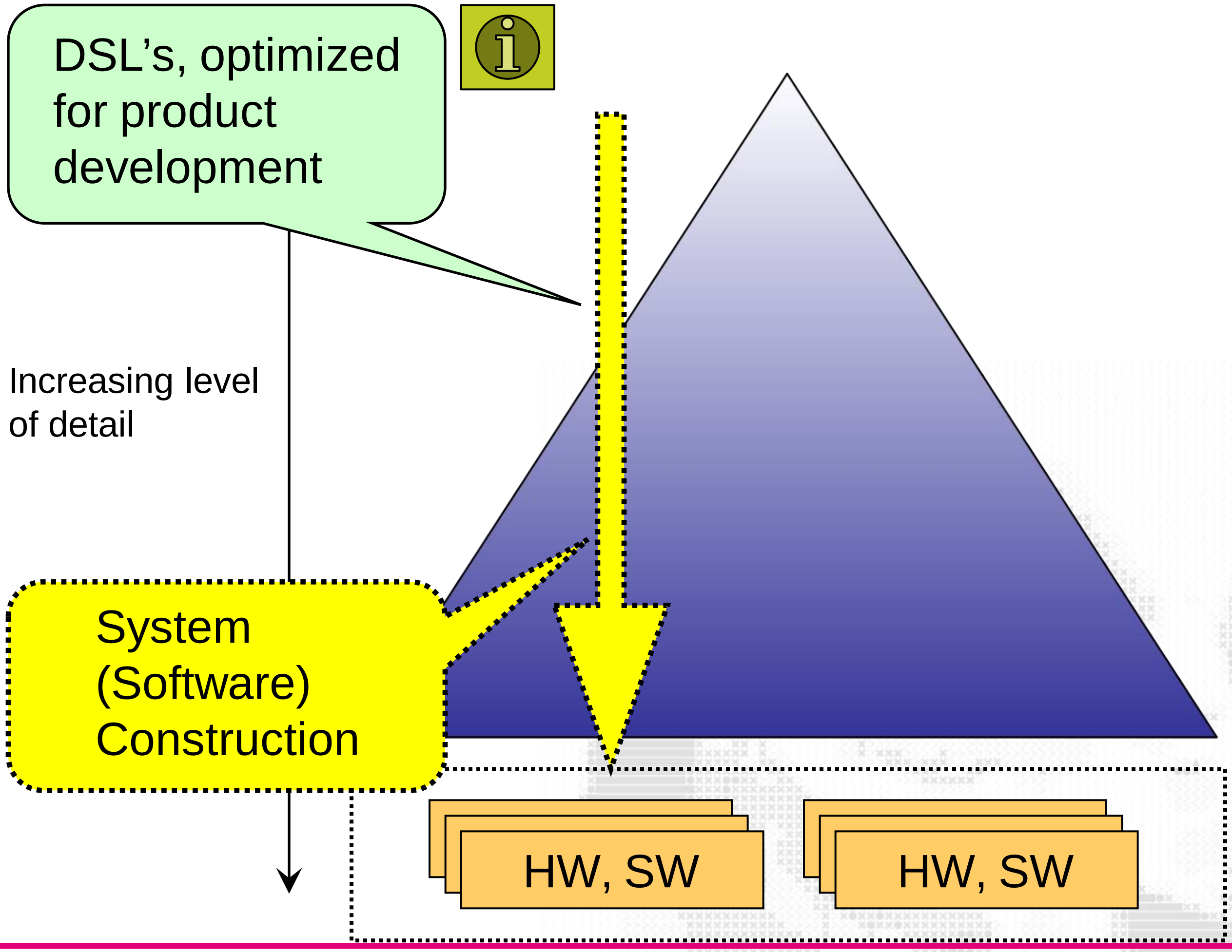
Software Engineering Should Be

- Rigorous Specification & Design: Model Driven Design using mathematics (formal verification) where possible

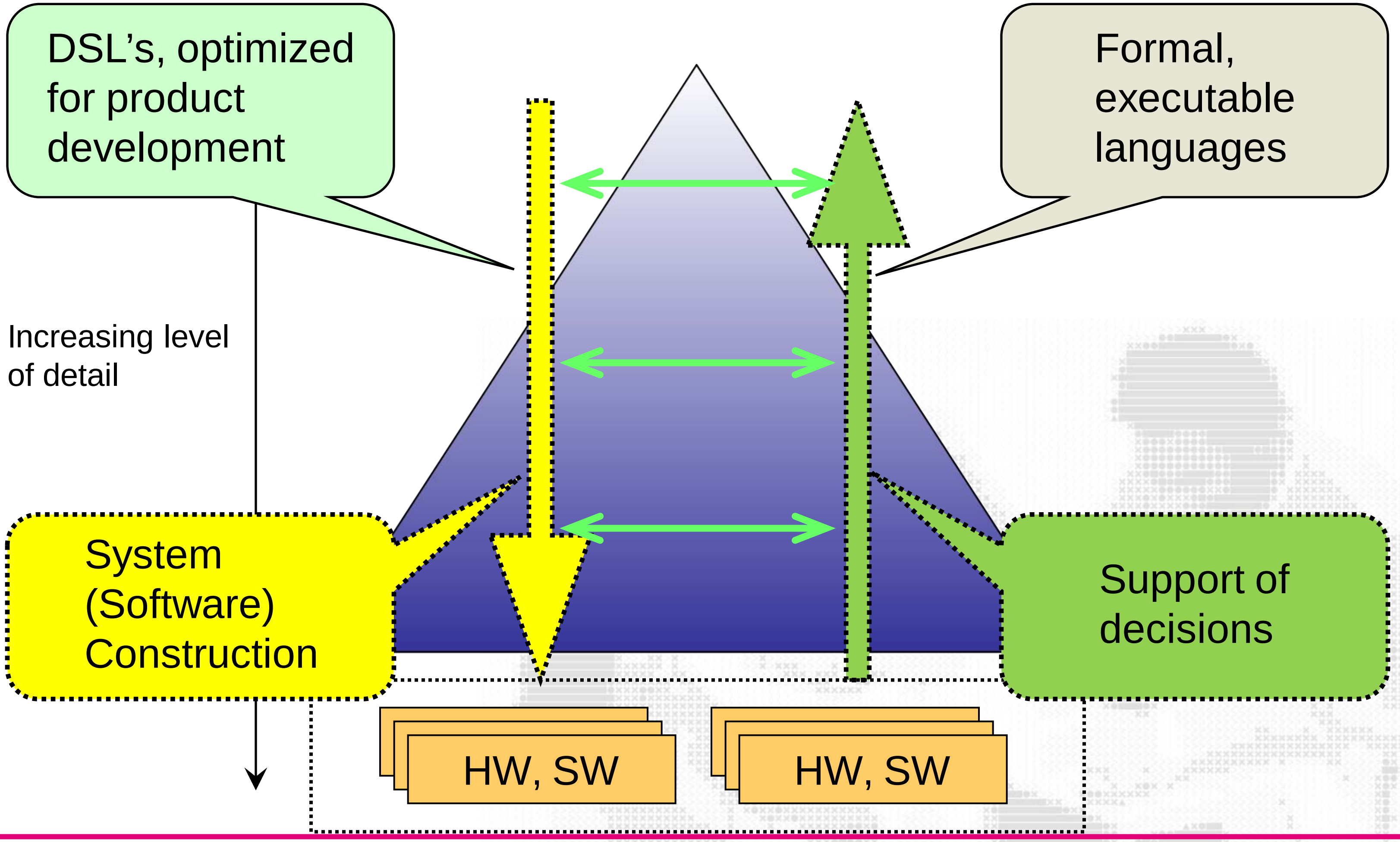


NSPYRE
making technology matter

Model Streams



Model Streams



NSPYRE

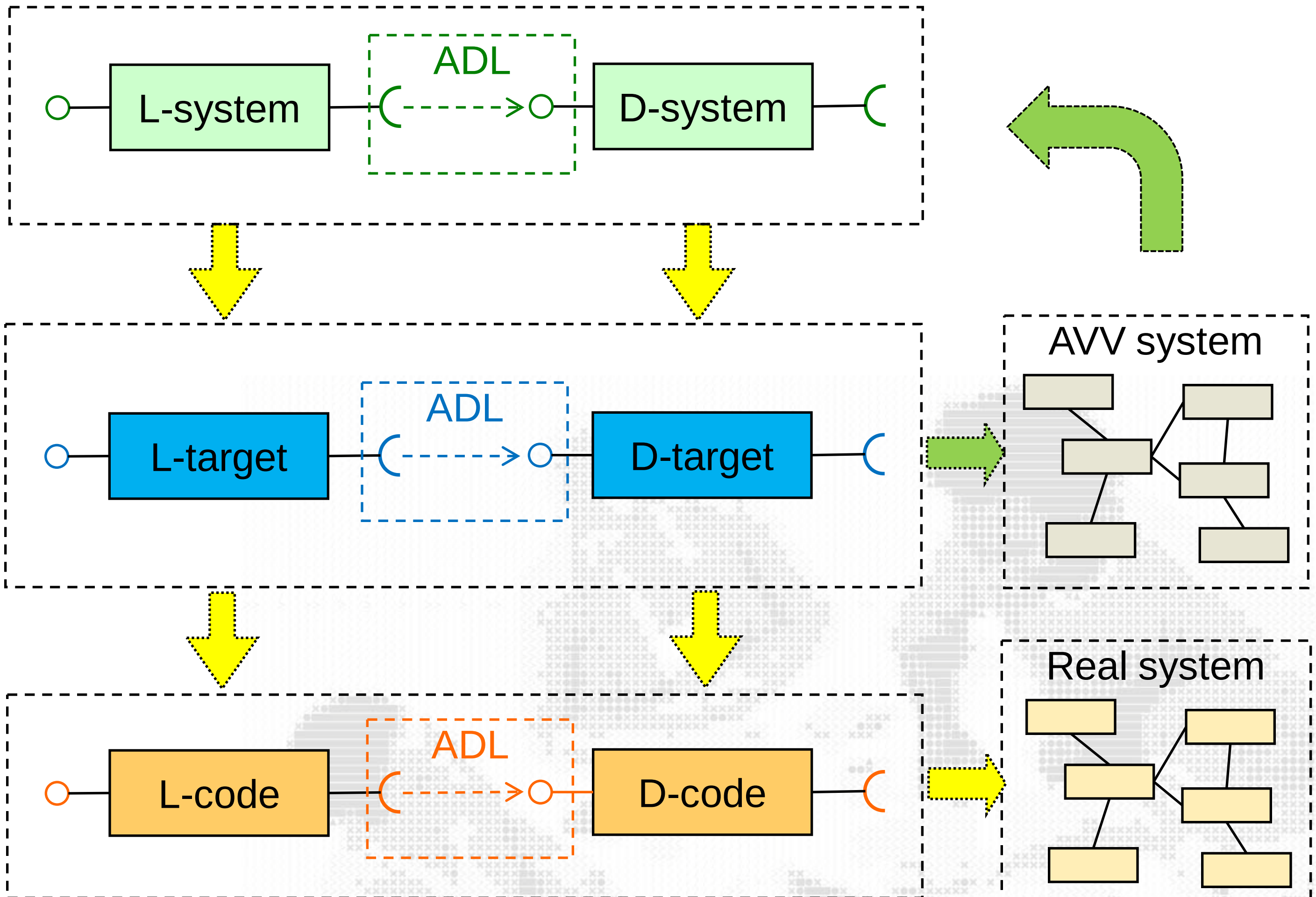
making technology matter

DSL Abstraction Levels

Essential level:
Specific DSL's
-Problem related
-Engineering efficiency
-Relate to problem domain

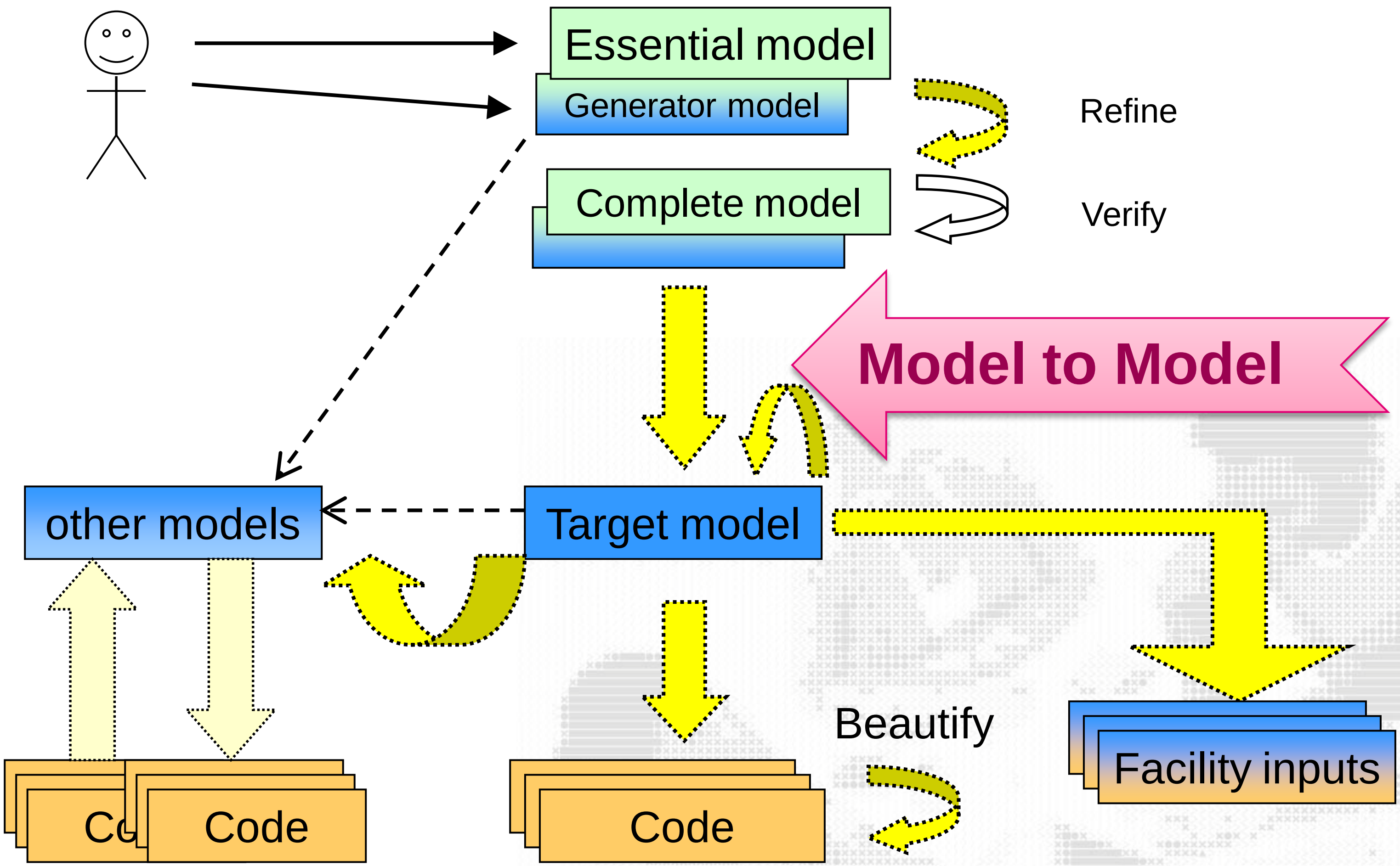
Intermediate level:
Generic DSL's
-Solution specific
-Early integration
-Relate to formalisms

Realization level:
GPL's

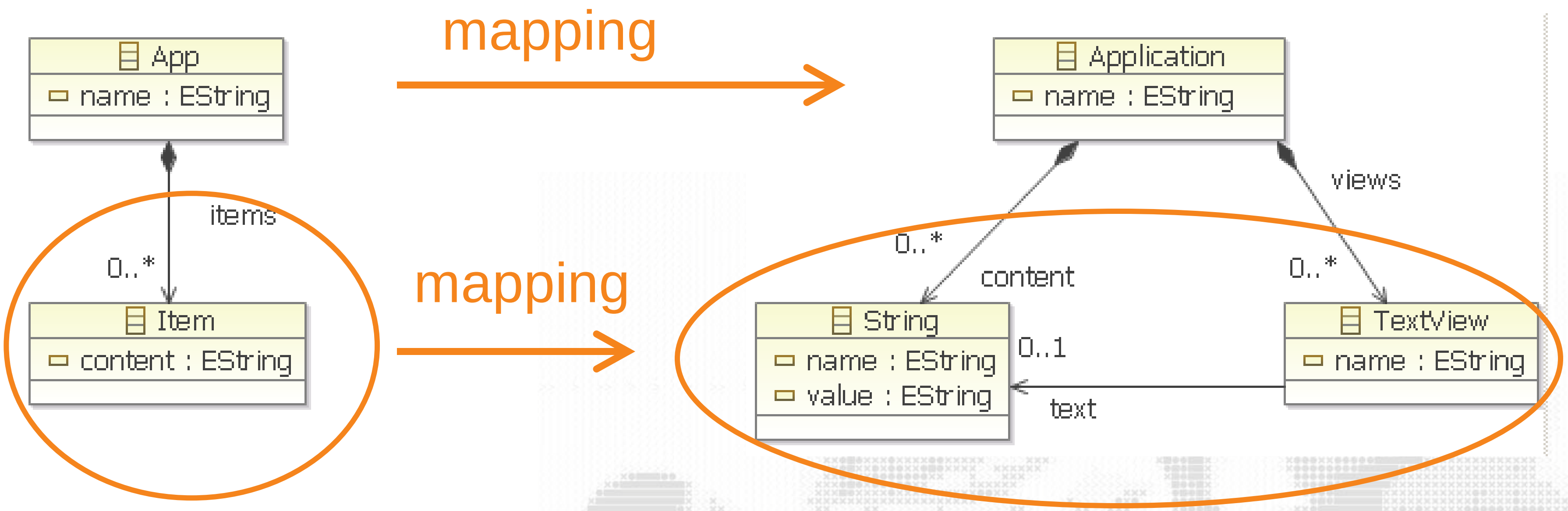


marc.hamilton@nspyre.nl

Generator Architecture: Overview



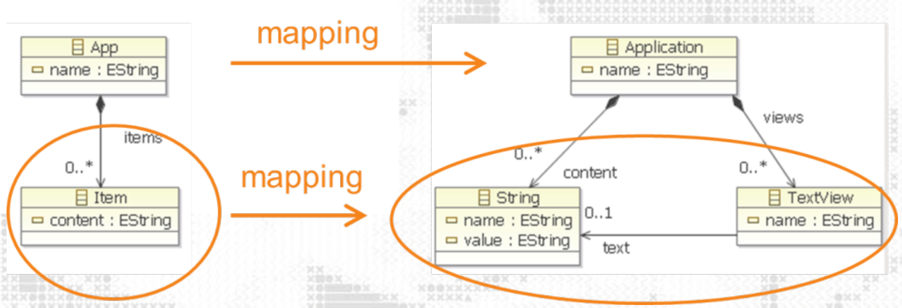
Model to Model



NSPYRE
making technology matter

Model to Model: QVTO transformation

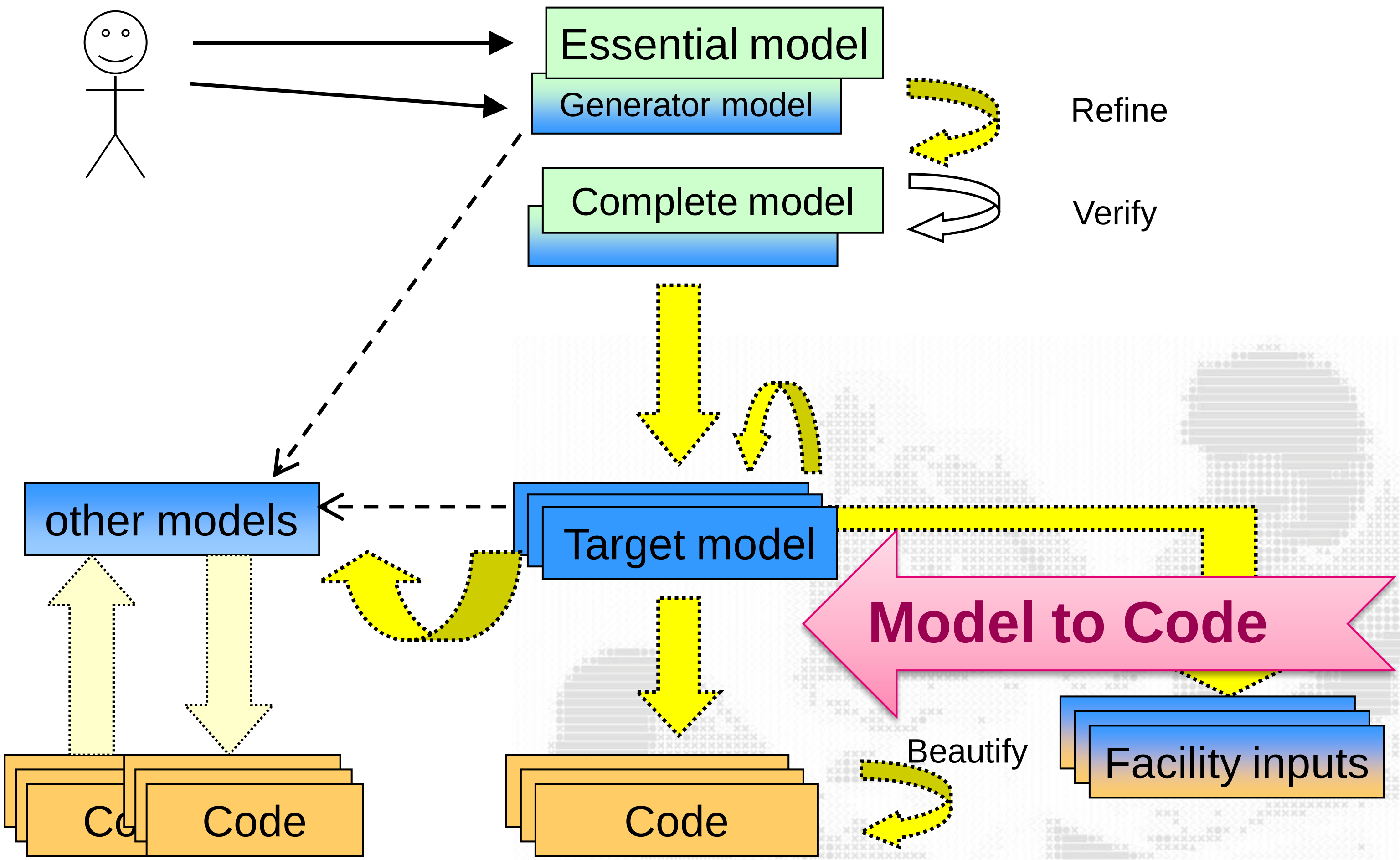
```
modeltype app "strict" uses myapp('http://myapp/1.0');
modeltype android "strict" uses myandroid('http://myandroid/1.0');

transformation App2Android(in appmodel : app, out androidmodel : android);

main() {
    appmodel.rootObjects()[app::App]->map App2Application();
}

mapping app::App::App2Application() : android::Application {
    name := self.name;
    content += self.items.map item2StringView(self).str;
    views += self.items.map item2StringView(self).tv;
}

mapping app::Item::item2StringView(in papp : app::App) : str: android::_String,
                                                         tv : android::TextView {
    init {
        var index := papp.items->indexOf(self);
        var string_name := 'string' + index.toString();
    }
    str.name := string_name;
    str.value := self.content;
    tv.text := str;
}
```



 marc.hamilton@nspyre.nl

making technology matter

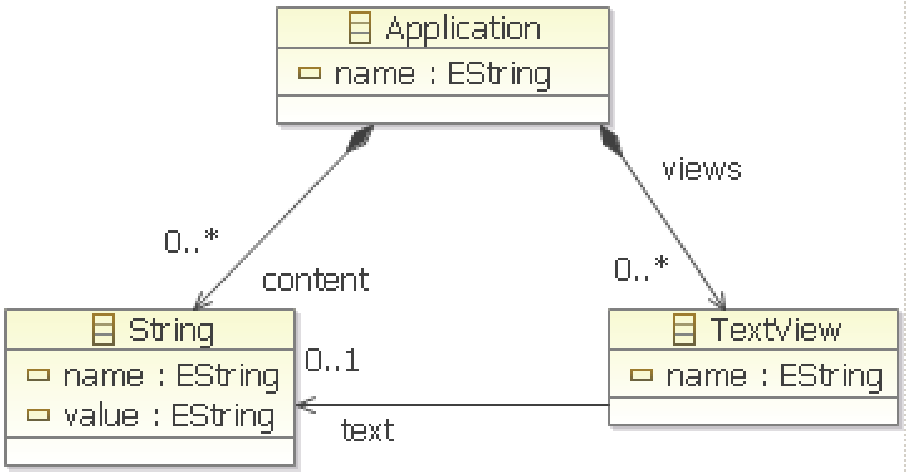
Generator Architecture: Overview



NSPYRE

making technology matter

Model to Code

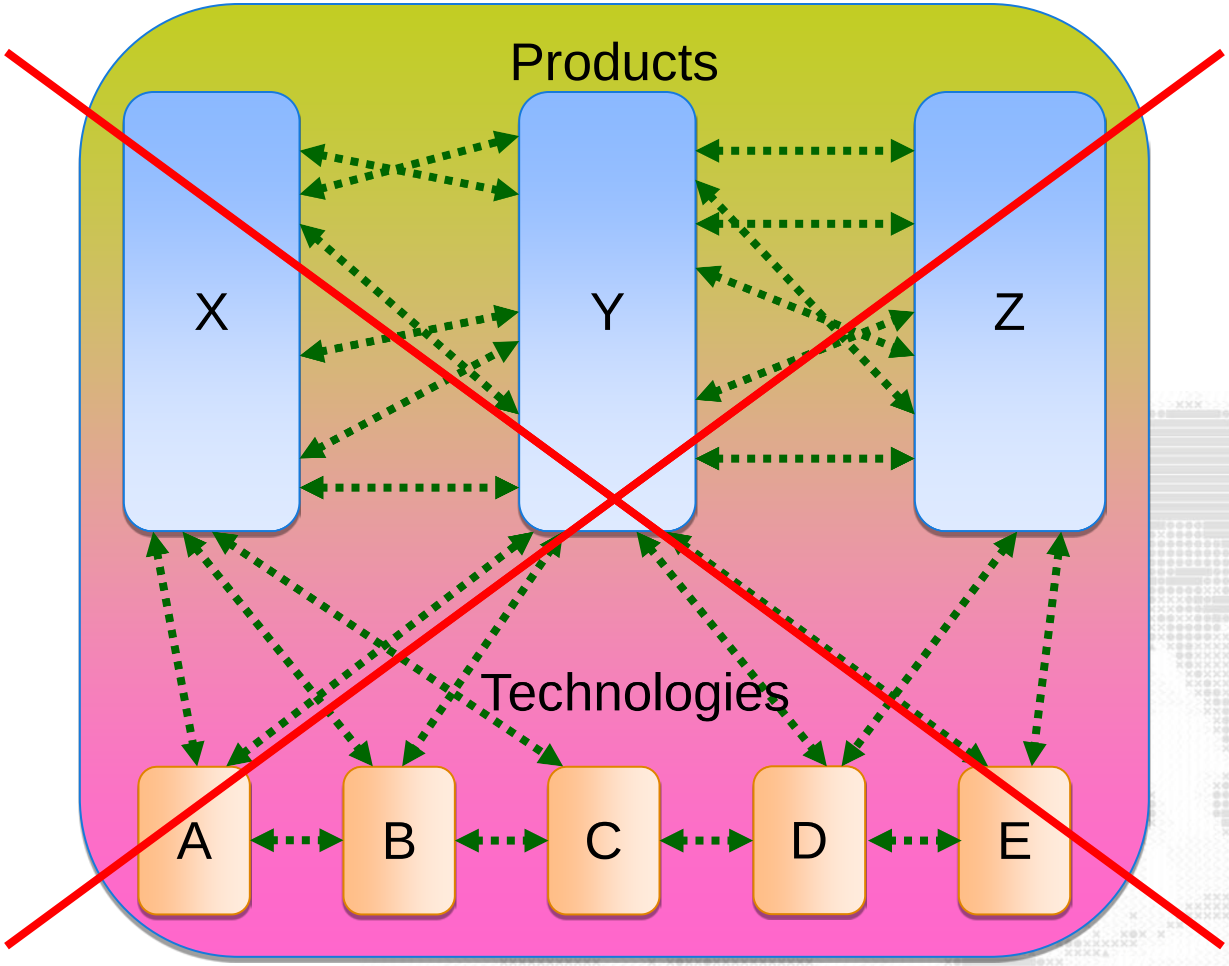



```
[template public generate_values(app : Application)]
    [file ('/res/values/strings.xml', false, 'Cp1252')]
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">[app.name/]</string>
    [for (s : String | app.content)]
    <string name="[s.name/]">[s.value/]</string>
    [/for]
</resources>
    [/file]
[/template]
```

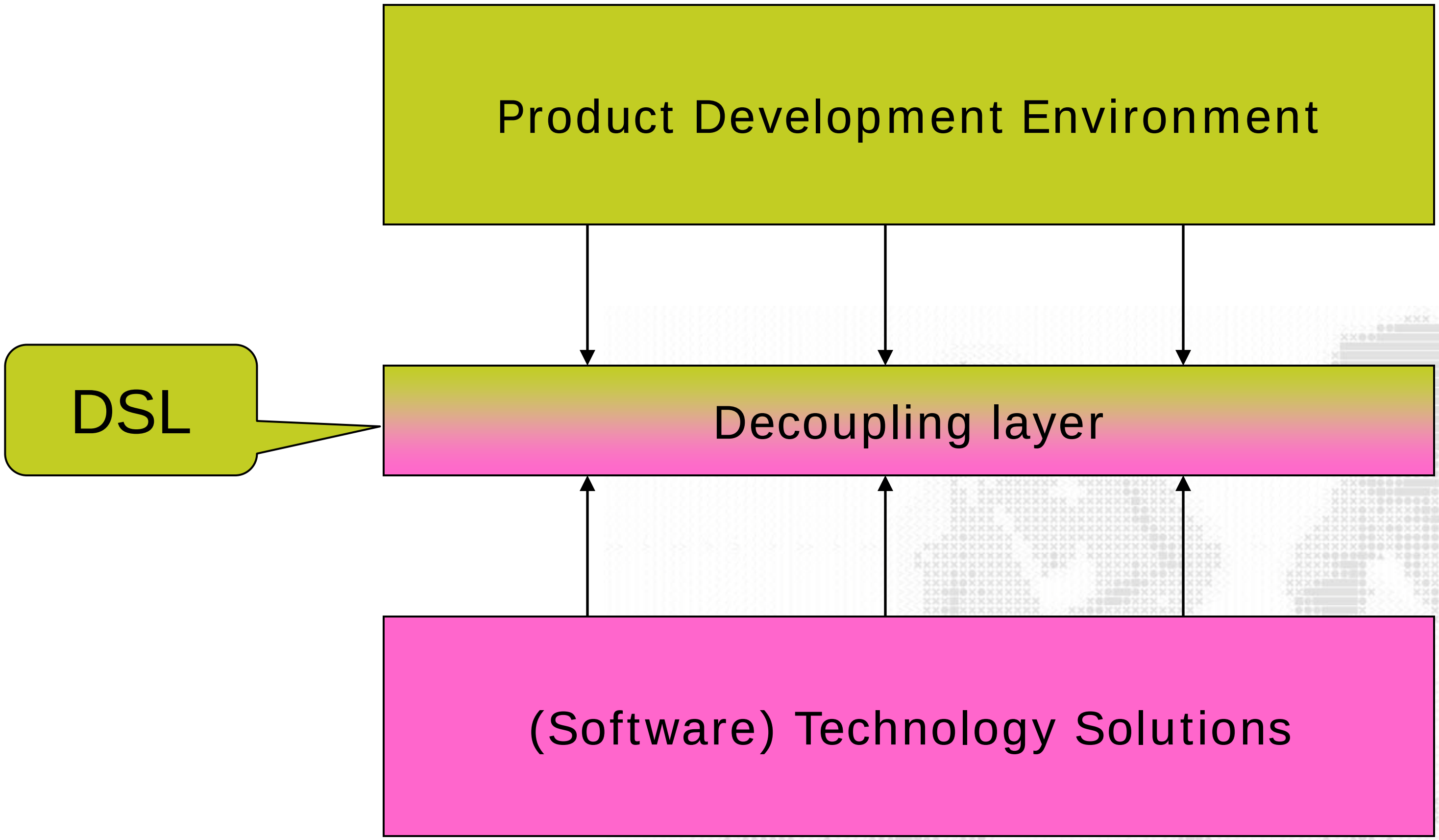
NSPYRE

making technology matter

Current Software Environment



Decoupling from Technology





NSPYRE

making technology matter

Industrialization: Factories!

Primary Process

Focus on building products in the factory

Secondary Process

Focus on building and maintaining the factory

NSPYRE
making technology matter

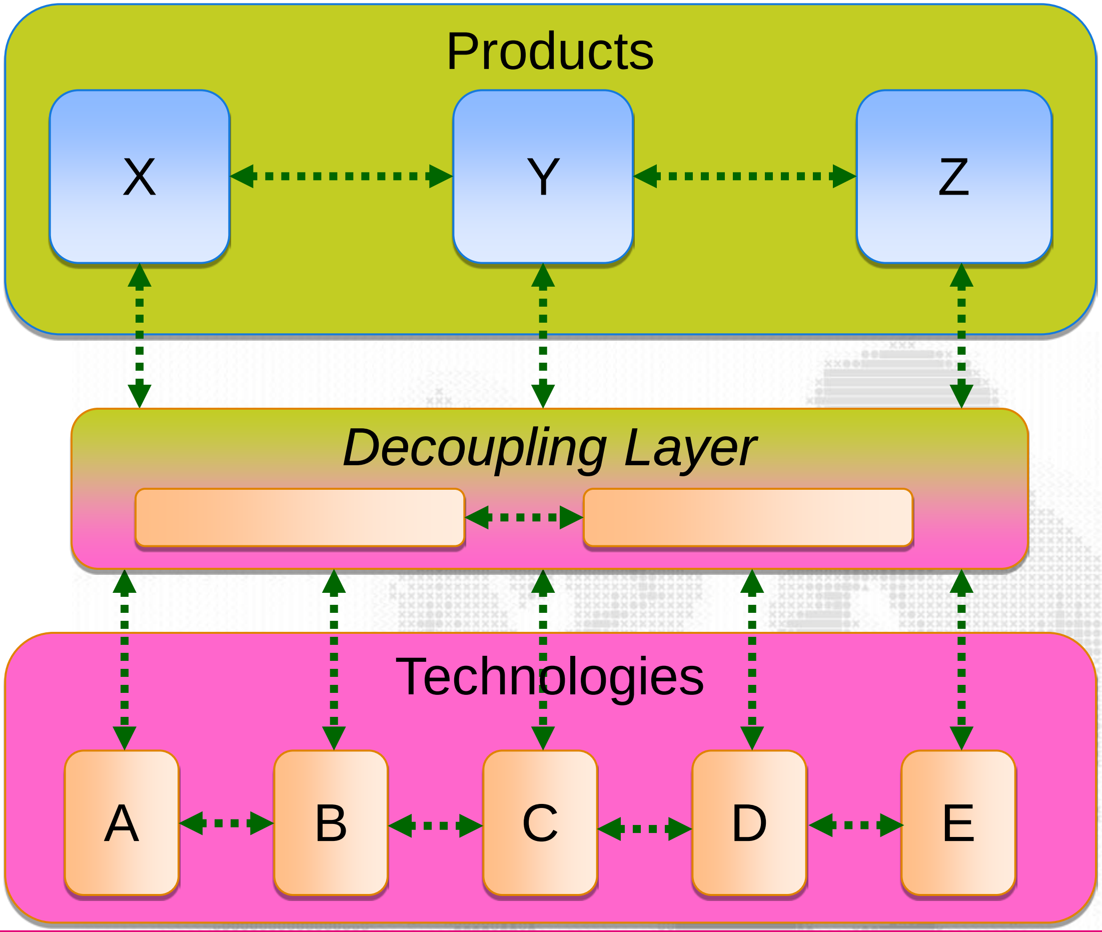
Software Factory!

Primary Process

Focus on building products in the factory

Secondary Process

Focus on building and maintaining the factory



NSPYRE

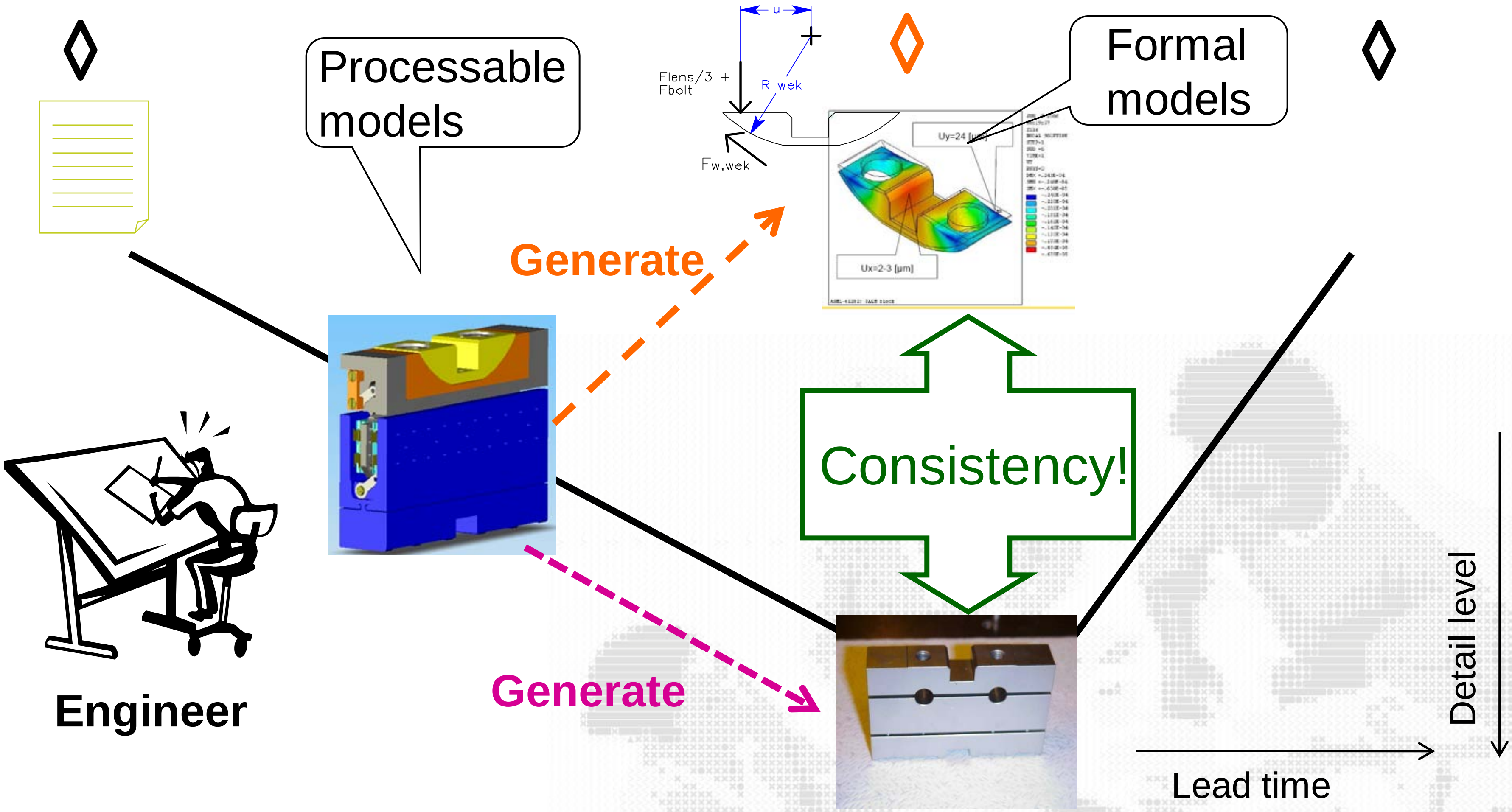
making technology matter

Vragen?

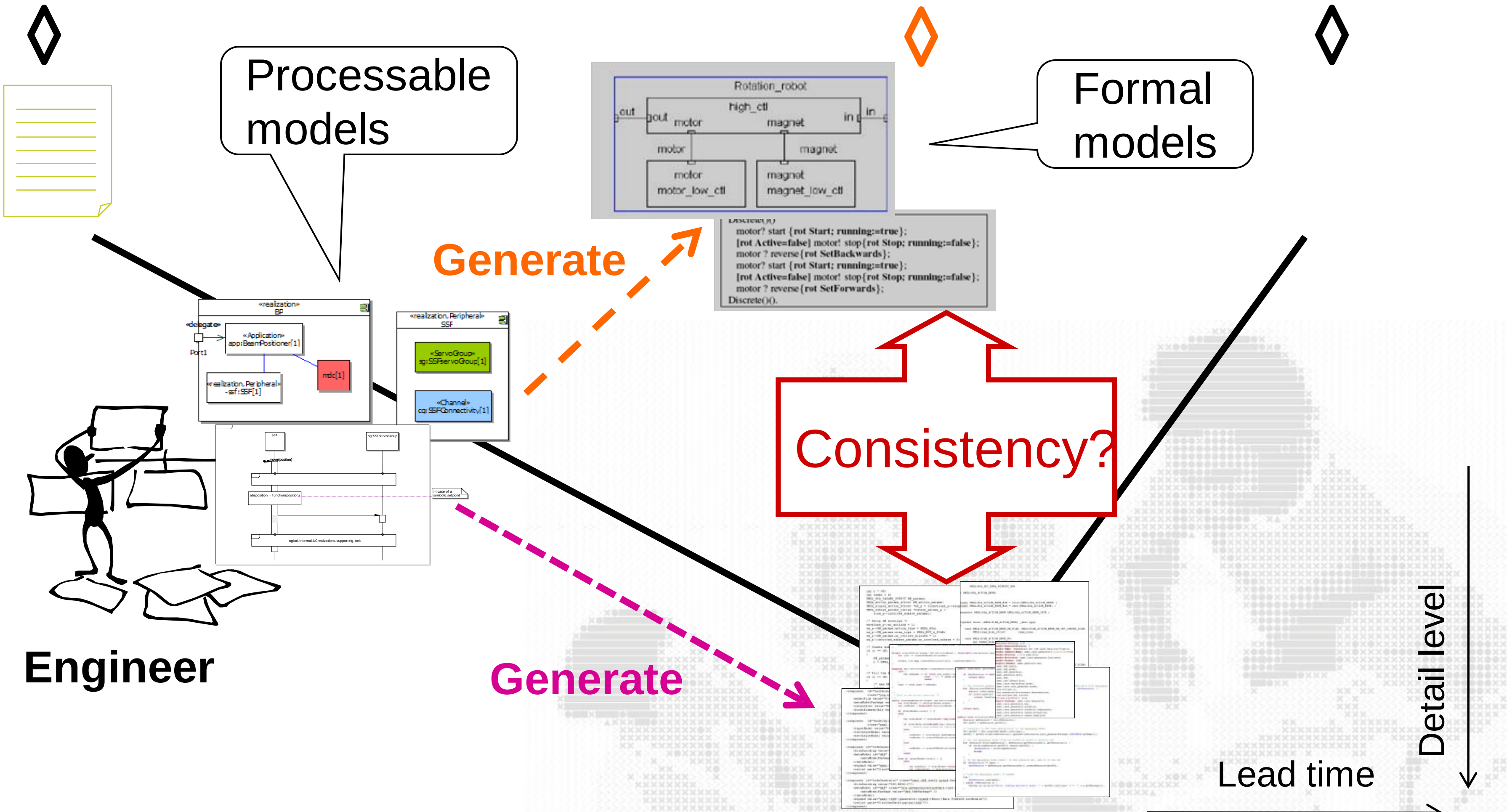
www.nspyre.nl

BACKUP

Mechanical Engineering



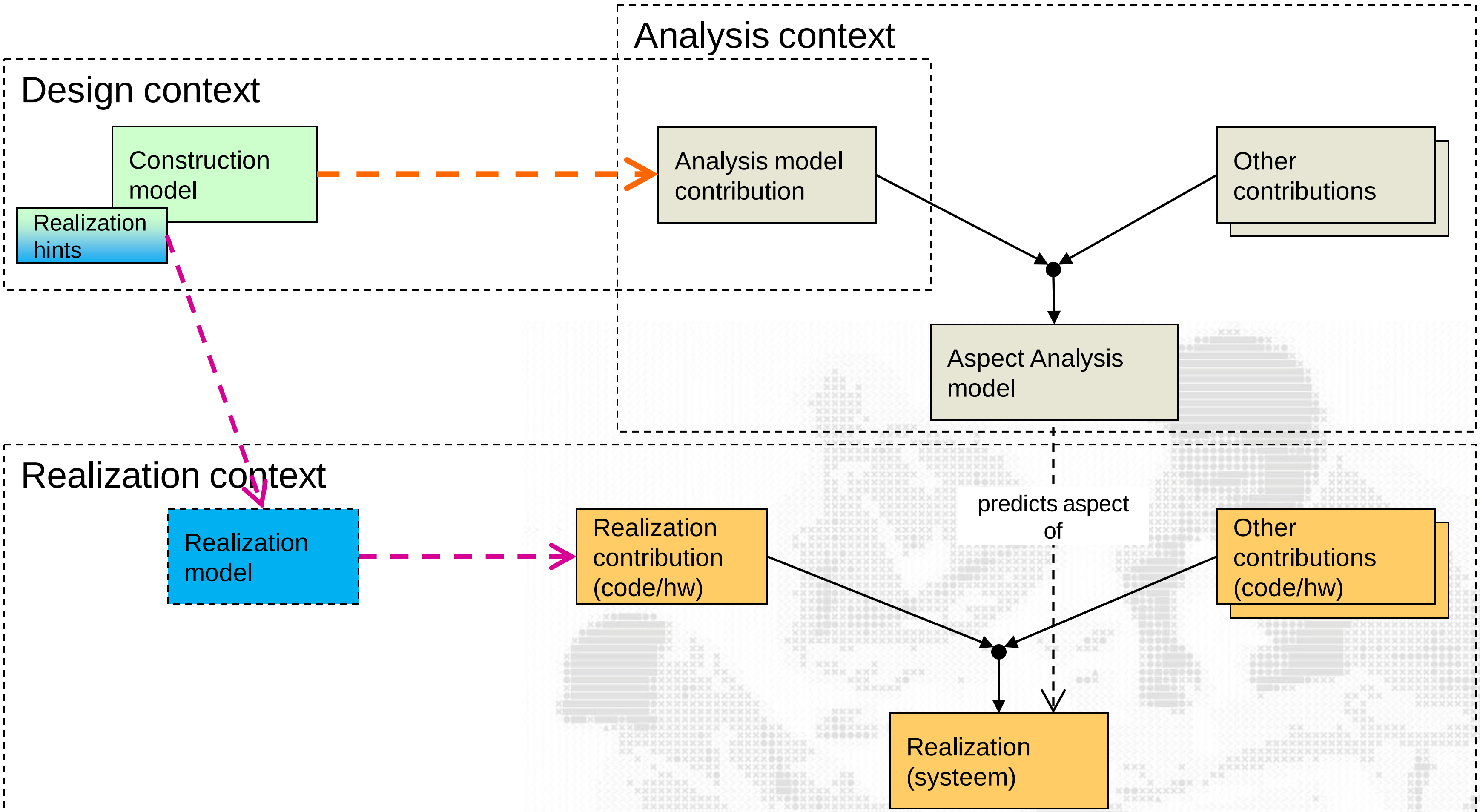
Engineering with models: Software MDE



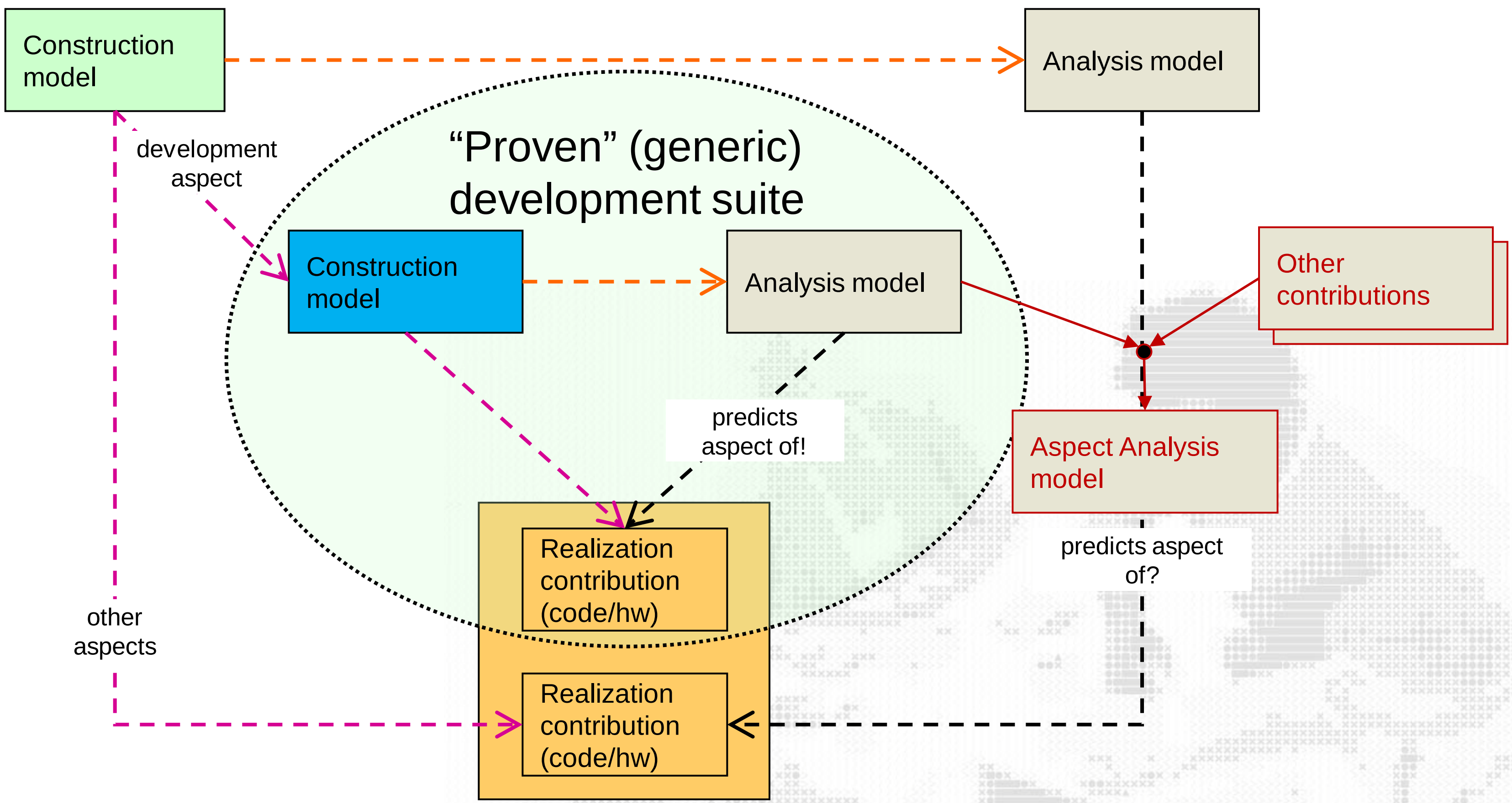
marc.hamilton@nspyre.nl

NSPYRE
making technology matter

Analysis Models

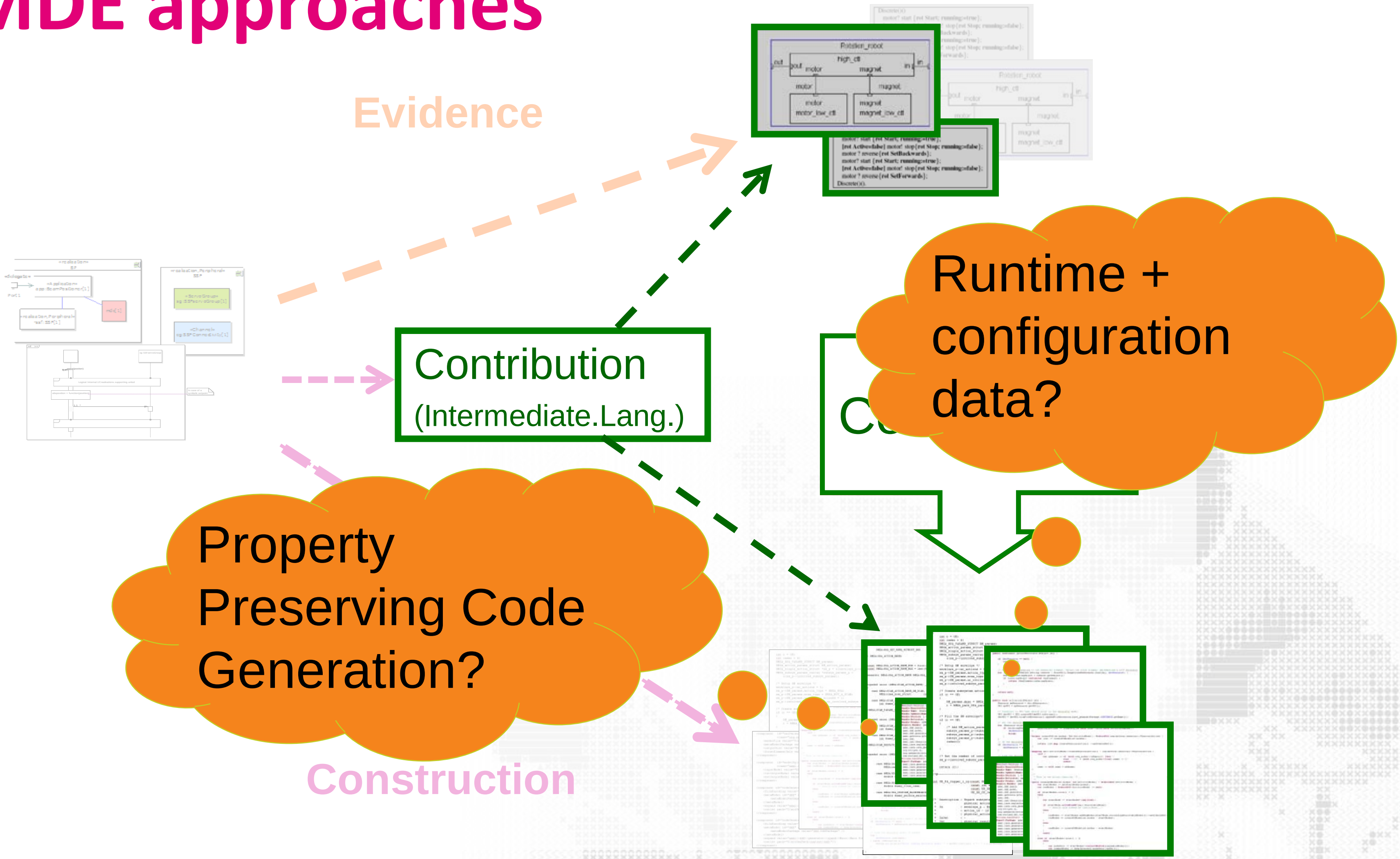


Embedding Proven Suites



NSPYRE
making technology matter

MDE approaches

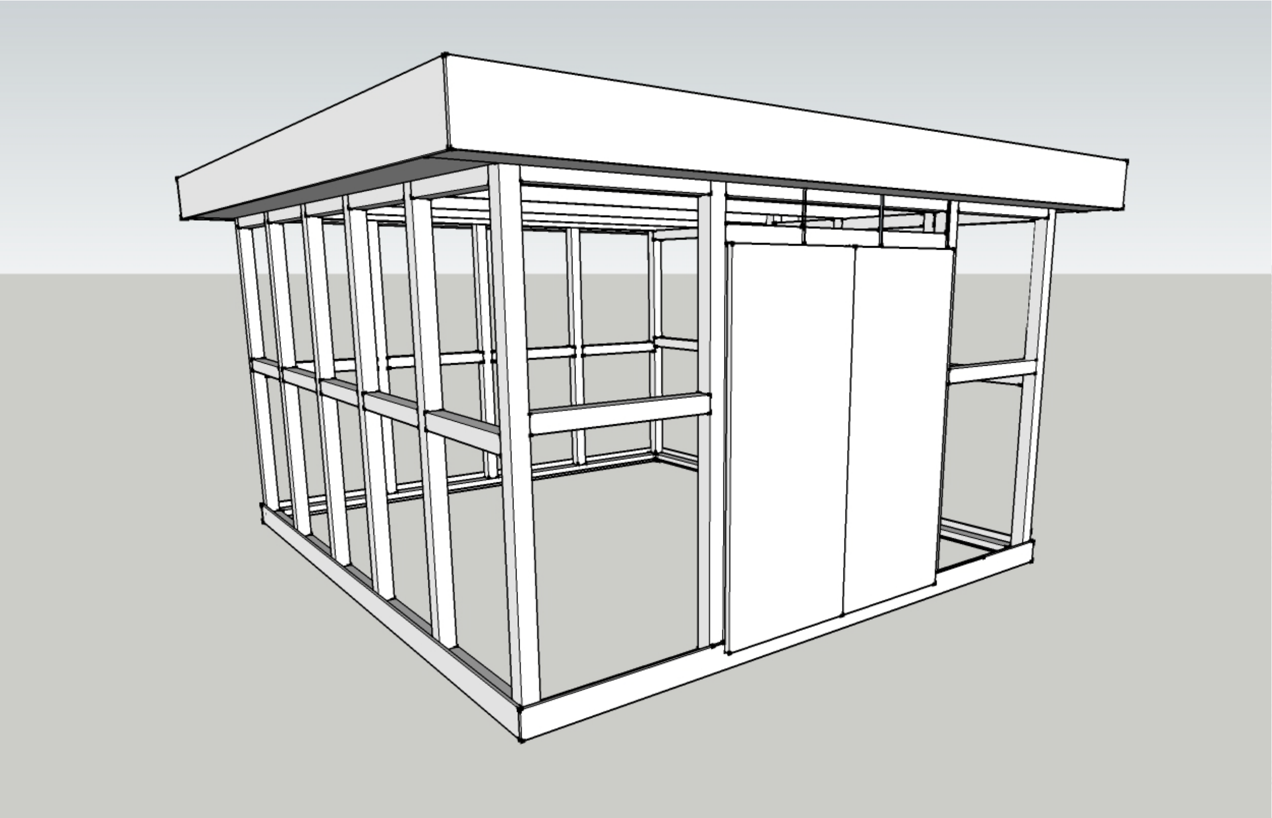


Tuinhuis (1) Garden shed

- “Blokhut” model: b x l x h = 4m x 4,5m x 2,5m.
- Houten frame 9cm op betonnen plaat 10cm.
- Dubbele deur aan voorkant: 1,40m

- Flat roof model: w x l x h = 4m x 4,5m x 2,5m.
- Wooden frame 9cm on concrete slab foundation
- Double door in front: 1,40m

Tuinhuis (2)



NSPYRE
making technology matter

Domain Specific Modeling

- Awareness of Concepts in the target domain
  - Meta-models define these concepts

➔ Domain Specific Language

marc.hamilton@nspyre.nl

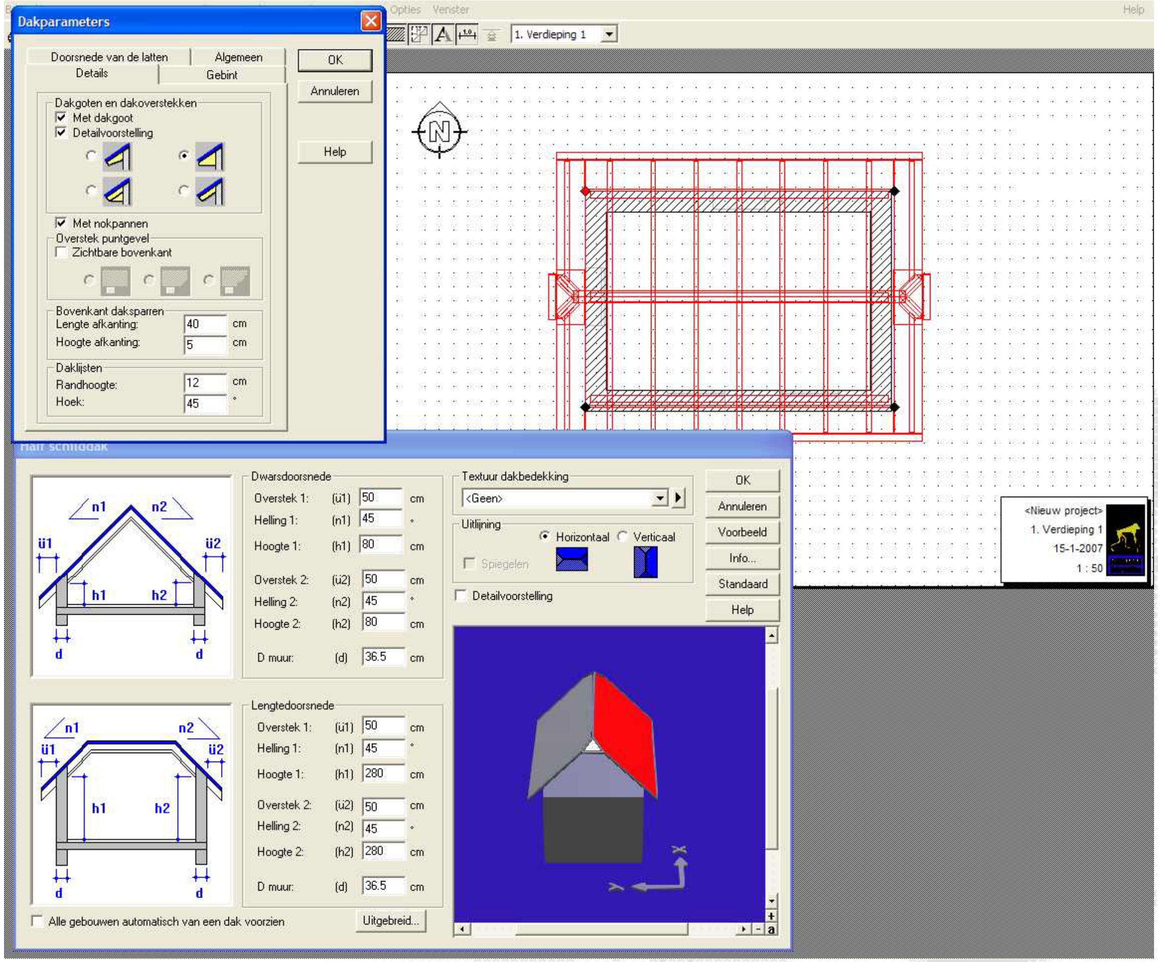
NSPYRE
making technology matter

Example: 3D Drawing

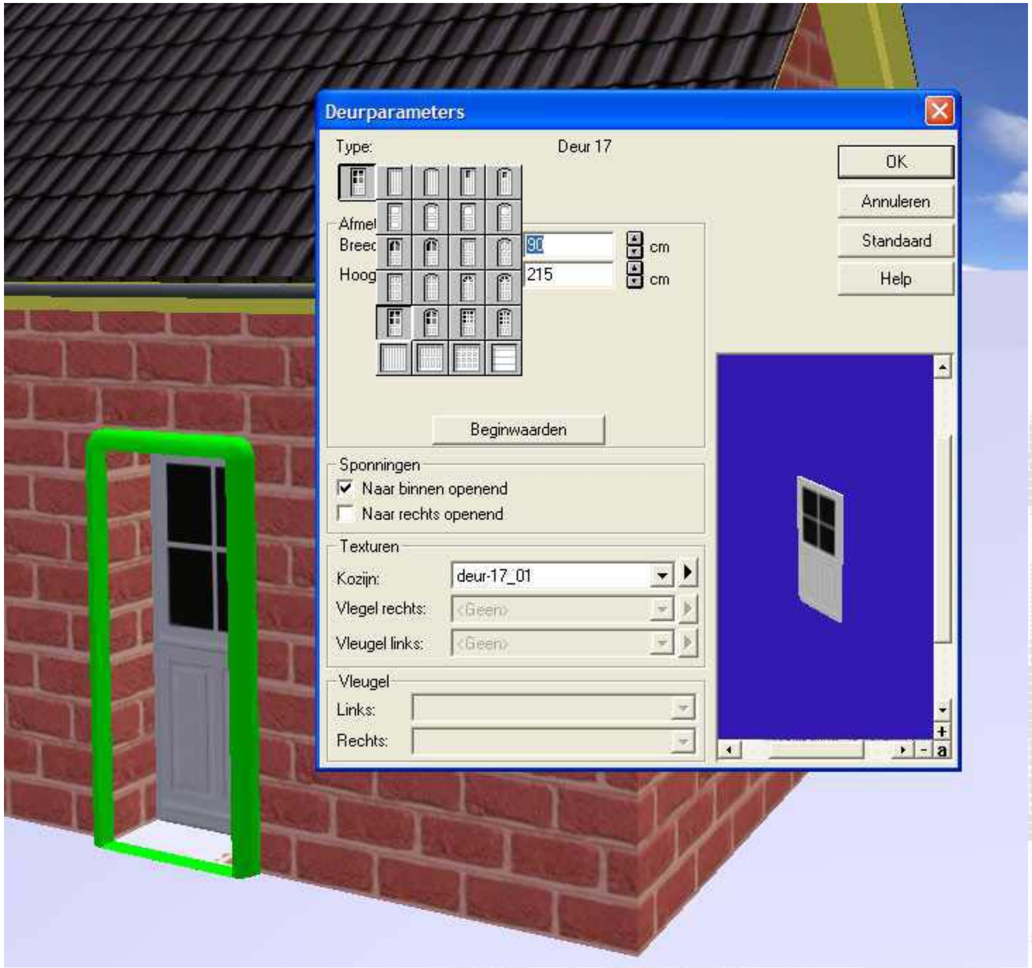
NSPYRE

making technology matter

Example: 3D Drawing



Example: 3D Drawing



Example: 3D Drawing