BESSER: BUILDING BETTER SMART SOFTWARE FASTER

Jordi Cabot
jordi.cabot@list.lu
Jordicabot.com

February 8th 2024
You’re asking too much!

Every time we master the art of building a type of software, new challenges pop up:

- Mobile dev
- Cyber-Physical systems
- Cloud-based and distributed systems
- Natural Language Interfaces
- AI
BECAUSE SOFTWARE DEVELOPMENT IS MORE COMPLEX THAN EVER
Chat with us

How can we help?

Welcome to my store

What can I do for you?
“Software is eating the world”
Andreessen Horowitz, HP (2011)

“Software is eating the world, but AI is going to eat software”
Jensen Huang, Nvidia CEO (2017)
SMART SOFTWARE SYSTEMS

Trad Software

AI component
Low-code modeling to the rescue
"Given the final model, the complete computerized information system can be automatically generated"

"we arrive at a specification from which executable code can be automatically generated"

Papers published at CAiSE’91

Let’s agree on low-code being a style of MDE that we use as companies easily buy into it
Low-code application platforms accelerate app delivery by dramatically reducing the amount of hand-coding required

– Forrester Report

Low-code is the evolution of previous model-based approaches (MDE, UML, MDA, …) where most of the application code is automatically generated from high-level designs of the system.
Growing market
The worldwide market for low-code is projected to $26.9 billion in 2023, a 19.6% increase from 2022.

Clients demand it
All major players (Amazon, Microsoft, Google,…) are adding low-code/no-code solutions to respond to this demand.

Low-code everywhere
Most software projects will ship with components developed with low-code tools in 2025.

Quality
Models can be tested and verified before starting the coding phase.

Productivity
Automation techniques take care of generating most of the system code (e.g. all the boilerplate code).

Portability
Model once deploy everywhere by just using different generators on top of the same model.
BUT LOW-CODE PLATFORMS ARE LIMITED TO DATA-ENTRY WEB APPS
SMART SOFTWARE SYSTEMS — EXISTING SOLUTIONS

Trad Software

I’ve your back

What??

AI component

You’re on your own
BESSER

Modeling to the rescue
GOAL: HELP COMPANIES MAKE BETTER SMART SOFTWARE FASTER
BESSER (BETTER SMART SOFTWARE FASTER) PROJECT

01. BESSER is a low-code approach to develop AI-enhanced software

02. BESSER will provide a single editor to model both traditional and smart components as well as possible ethical concerns

03. BESSER will provide tools to generate and test all modeled components altogether (e.g. a chatbot, the web UI, the database, a prediction system...)

04. BESSER is a 5-year FNR PEARL project.

05. The core BESSER components will be released as OSS.

06. Open to external collaborations. Commercial extensions will be allowed.
WE NEED (AND WILL PROVIDE)...

ML DSLs

(Semi)automate the generation of smart software

Better models for smart software

Better ways to create such models

Code-generation

Low-modeling
DSLS TO MODEL SMART MODELS

(not all of them already part of the platform)
A social artifact that is acknowledged by an observer to represent an abstraction of some domain for a particular purpose. – E. Proper & G. Guizzardi

Everything is a model – J. Bézivin
ML MODELS ARE MODELS

• For instance, you can see them as model transformations that generate output data from their input

• Many ML operations (e.g. fine tuning) can be mapped to “classical” model manipulation operations such as model refining

MDE techniques would then be useful for AI people!? . E.g. what about ML model merge (getting very popular right now). How can we model the merge? And/or the selection of models to be merged?
All major vendors are “going modeling”
LangFlow is a simple workflow modeling environment for LangChain
Not only modeling -> also MDD

We’re on a mission to unify all ML frameworks 🌐 + automate code conversions 🌐. pip install ivy-core 🌐, join our growing community 🤗, and lets-unify.ai 🤗.
• Ongoing concerns from the AI community to clarify the representativeness of the data, its provenance, possible social issues,…. But no industry standards for documenting ML datasets

• Our solution: DescribeML -> A DSL to describe ML datasets

• Enabling a precise definition of provenance, distribution, annotation process, potential biases… of a dataset
**DSL METAMODEL**

- **Metadata**
  - `uniqueId`: String
  - `title`: String
  - `version`: Version
  - `dates`: Dates
  - `description`: Description
  - `tags`: Tags
  - `licenses`: LicensesEnum
  - `applications`: Applications
  - `authoring`: Authoring

- **Composition**
  - `compositionRationale`: String
  - `dataInstances` (1..*)

- **DataInstance**
  - `id`: String
  - `description`: String
  - `type`: TypeEnum
  - `size`: Integer
  - `statistics`: InstanceStatistics
  - `rules`: ConsistencyRules
  - `attributes` (1..*)

- **Attribute**
  - `id`: String
  - `description`: String
  - `type`: AttrTypeEnum
  - `statistics`: AttributeStatistics

- **LabelingProcess**
  - `description`: String
  - `type`: LabelingEnum
  - `requirements`: Requirement
  - `demographics`: Demographics

- **GatheringProcess**
  - `description`: String
  - `type`: GatherEnum
  - `sources`: Sources
  - `demographics`: Demographics
  - `requirements`: Requirement
  - `labelingProcess` (0..*)

- **Provenance**
  - `curationRationale`: String
  - `gatheringProcesses` (0..*)

- **SocialConcerns**
  - `generalRationale`: String
  - `issues` (0..*)

- **SocialIssue**
  - `description`: String
  - `type`: IssueTypeEnum
  - `relatedAttributes`
• Plug-in for VSCode

• Implemented with Langium
EXAMPLE DSL2 — IMPROMPTU, A STRUCTURED LANGUAGE FOR BETTER / REUSABLE / INDEPENDENT PROMPTS

```python
prompt GenerateImage(
    @animal "Name of an animal"): image
    core = "Image of " @animal " in its natural habitat"
    traits = quality(high), art_form(photography)
)

prompt ImageQuestions(
    $image "Image",
    @animal "Animal appearing in the image",
    @level "English level"): text
    core = "Propose 5 questions for an English course regarding the picture of", @animal
    suffix = Exam.DetailedInstructions(@level)
    composer CompleteExam(
        @grammar "Grammar questions",
        @questions "Questions about the image")
    "Answer the following questions:
    @grammar, "(50%)
    @questions, "(50%"
)
```
ONE GOAL, UNDERSTANDING THE INTENTION

• Chat/VoiceBots are created with the BESSER Bot Language offering a bot-specific syntax for creating:
  ▪ Intents the bot needs to match
  ▪ The behavior to execute in response to the matched intents

• The language is designed to easily integrate and combine all types of IUls
DEFINING THE BOT’S LANGUAGE: INTENTS

• Intent: Goal or purpose behind message or query
• Bot needs to be aware of the possible intents it should be able to handle
• Bot requires example sentences for each intent to know what to look for
• Supported languages:
  • English, French, German, Spanish, Catalan, (partial) Luxembourgish and more!

```python
hello_intent = bot.new_intent('hello_intent', [
  'hello',
  'hi',
])
```
## Defining States and Their Content

```python
# STATES

initial_state = bot.new_state('initial_state', initial=True)
hello_state = bot.new_state('hello_state')
good_state = bot.new_state('good_state')
bad_state = bot.new_state('bad_state')

# HELLO BODY

def hello_body(session: Session):
    session.reply('Hi! How are you?')
    session.set("message", session.message)
    # Anything can be done here!

hello_state.set_body(hello_body)
```

Diagram:

- **initial_state**
  - **hello_intent**
    - **User:** "hello"
    - **Bot:** "Hi! how are you?"
  - **good_intent**
    - **User:** "good"
    - **Bot:** "I am glad to hear that!"
  - **bad_intent**
    - **User:** "bad"
    - **Bot:** "I am sorry to hear that..."

- **good_state**
- **bad_state**
Transitions are caused by events

- Events caused by the user

**Intent events:**
- Match recognized intent from user message to defined intent in transition definition

**Other events:**
- Transition based on stored session value
- Automatic transitions
BOTS / AGENTS FOR OSS SUSTAINABILITY

- Open Source Software Sustainability
- Bots in OSS
PROPOSAL

**Tool Infrastructure**
- **DSL Definition**
- ** Helpers**

**Runtime**
- **Event Listener**
- **Connectors**

**GitHub**

Connections:
- **uses**
- **conforms to**
- **triggers**
- **calls**
- **interacts**
- **notifies**
LOW-MODELING
Smart software is a complex beast, we need models to tame this essential and accidental complexity.

Every day there are more things to model!!!!!
To model, or not to model, this is the WRONG question

Real question is the ROI of making the models explicit

- Shakespeare
Low-code application platforms accelerate app delivery by dramatically reducing the amount of hand modeling required. Forrester Report.
LOW-MODELING IS NOT BRAND NEW EITHER

Rule 1

Input Model
- Document
- Conference

[L] Author paper list
- <<List>> Paper list
- <<DataBinding>> Paper

Output Model
- Conference
  + createConference()
  + deleteConference()
- Document

Paper details
- <<Details>> Paper details
- <<DataBinding>> Paper

Coauthor list
- <<List>> <<DataBinding>> Author

Review list
- <<List>> <<DataBinding>> Review

Subject
- id: decimal
- name: string

Paper
- id: integer
- title: string
- abs: text

Track
- id: decimal
- name: string
- discription: string

(a)

(b)
AI GIVES MODELERS SUPERPOWERS
I tried (and fail) to write this myself 10yrs ago. ChatGPT does it with no additional training. But it doesn’t like unnatural models
• Going beyond one-shot “text-to-model” transformations
• Goal is to have a conversation that helps confirm / complete partial models being created by the chatbot
• There is a hidden model in your database, in your API or web forms. Or your CSVs
• Even in your docs, manuals and regulations

• How far can we push AI technologies to make sense of the implicit models behind all these data?
Empowering citizens to benefit from open data sources (>1.6M in the European data portal)

Partial Bot models are generated from the CSV/JSON file

- E.g. checking the type of the columns we can generate obvious questions (\textit{max}, \textit{avg}, \textit{min} for numeric ones)
- Ontologies could be used to package more semantic libraries of questions
- TextToSQL used as default fallback
BOTS FOR OPEN DATA PROJECT

Data description inference → Data description → Bot generation

Synonyms, translations, ...

Data description enhancement (optional) → Enriched data description

OpenAI
BIG PICTURE
THE BESSLER PROJECT

- Collects: Documents, DBs & Other Org Knowledge available
- Specifies: Domain Knowledge, Goals & Ethical concerns
- Provides: Data
- Receives: Explanations, Quality dashboard

**Smart software model**
- Traditional SW model
- Smart back-end
- Smart front-end
- AI quality properties

**BEsser DSLs & Smart Editors**

**Smart Low-code Generators and run-time Infrastructure**
- Training component: Code, Data, Params
- ML Model
- Deployment component: AI Platforms
  - Local packages
  - Cloud services
- Code Generation

**BEsser Low-modeling**
- Structured Knowledge Importer
- Unstructured Knowledge Importer

**Monitoring**, **Testing**, **Explainability**
STATUS
As of version 0.1 (open-source, already available on GitHub)

B-UML Language

Code Generators

- SQLAlchemy
- Py Domain Model
- Django models
- Other Generators

Generated Code

https://github.com/besser-pearl
BESSER LOW CODE — DESIGN A MODEL USING PLANTUML

PlantUML to B-UML Model

```plaintext
@startuml
class Library {
    + name: str
    + address: str
}

class Book {
    + title: str
    + pages: int
    + release: date
}

class Author {
    + name: str
    + email: str
}

Book "*" -- "1..*" Author: writtenBy
Library "1" -- "*" Book: has
@enduml
```
BESSER LOW CODE — DESIGN A MODEL USING PYTHON

Defining the B-UML model using Python

```python
from Buml.metamodelstructuralstructural import DomainModel, Class, Property, 
    PrimitiveDataType, Multiplicity, BinaryAssociation

# Library attributes definition
library_name: Property = Property(name="name", owner=None, property_type=PrimitiveDataType("str"),
    address: Property = Property(name="address", owner=None, property_type=PrimitiveDataType("str"))
# Library class definition
library: Class = Class (name="Library", attributes={library_name, address})

# Book attributes definition
title: Property = Property(name="title", owner=None, property_type=PrimitiveDataType("str"),
    pages: Property = Property(name="pages", owner=None, property_type=PrimitiveDataType("int"),
    release: Property = Property(name="release", owner=None, property_type=PrimitiveDataType("date"),
# Book class definition
book: Class = Class (name="Book", attributes={title, pages, release})

# Author attributes definition
author_name: Property = Property(name="name", owner=None, property_type=PrimitiveDataType("str"),
    email: Property = Property(name="email", owner=None, property_type=PrimitiveDataType("str"),
# Author class definition
author: Class = Class (name="Author", attributes={author_name, email})
```
BESSER LOW CODE — DESIGN A MODEL USING AN IMAGE

From an image to B-UML

B-UML Language

B-UML

MM

Instance of

Model

PlantUML
Python
Hand-draw

B-UML

Model

Library
name: str
address: str

Book
Title: str
pages: int
release: date

Author
name: str
eMail: str

1..* unlinked by

2 has

B-UML

Model
BESSER LOW CODE – CODE GENERATORS

B-UML Model → Code Generation → Django Web App using the generated code
Besser – Smart Code Generators

OpenAI

Code Generation template

B-uml Model

Django Web App using the generated code

Django administration

Add book

AUTHENTICATION AND AUTHORIZATION
Groups + Add
Users + Add

BOOKS
Authors + Add
Books + Add
Libraries + Add

Page:

Release:
Date: [ ] Today
Time: [ ] Now

Title:

Author:

Library:

Save Save and add another Save and continue editing
SubTyping

Digital Product Passport (DPP)

Climaborough

Promotion transformation (by creating Python classes from the model element instances)
• State machines via the integration with the BESSER bot framework

• Architectural DSL

• Web and mobile UI DSL

• OCL support
OFFER A NO-CODE PATH (THOUGH NO CLEARCUT BETWEEN THE TWO)

There is some no-code in low-code and some low-code in no-code
OK, BUT WHY YET ANOTHER TOOL?

(Honest answer for my fellow researchers)
WHAT ABOUT ECLIPSE?

- Eclipse is not a sexy IDE anymore
- Unclear who maintains EMF and/or what we can expect out of it in the future
- The world of ML revolves around Python

CAUTION: I bet against Eclipse before (in 2003!!) – anybody remembers Netbeans MDR?
Easier also at the ORG level (IP Concerns), e.g. we even reimplemented Xatkit.
ARCHITECTURAL DECISIONS
• External DSLs are great until you realize you are reinventing Java

• If your external DSL needs conditionals, loops, calls to external APIs,… move to an Internal DSL

• Fluent APIs are quite good at giving the impression to your users that they are using a “real” DSL
ANTLR FOR CONCRETE SYNTAX

- No Good “concrete syntax to my own Metamodel solution”

- We played with TextX to stay in the “Xtext ecosystem”

- But we finally went for a “raw” solution. ANTLR to get the AST and from there we write a model transformation to create the abstract syntax instance

- The model (and not any syntax) is the only truth
What do I want from you?
*Sorry Francis, I couldn't help it, this slide had to appear in your honor
**btw, don't worry for Papyrus, it seems it has resurrected
I’d love to get your feedback.

Of course, I’d even love more that you test the tool and even contribute it (but I’m realistic)
Very inspirational talk, do you recommend me to start my own tool?
Better contribute to existing initiatives, like ours ;-)

NOOOOOOOOOOOO!
DIFFICULT TO GET A GOOD RESEARCH ENVIRONMENT TO START A LONG-TERM TOOL BUILDING INITIATIVE

Stable and generous funding

Better recognition for tool development (e.g. DORA)

More venues for tool papers

New team that you can shape and agree on the goal

Evaluated on impact not just papers

Being an RTO (companies expect real tools)

LLM
Questions?

Jordi.cabot@list.lu