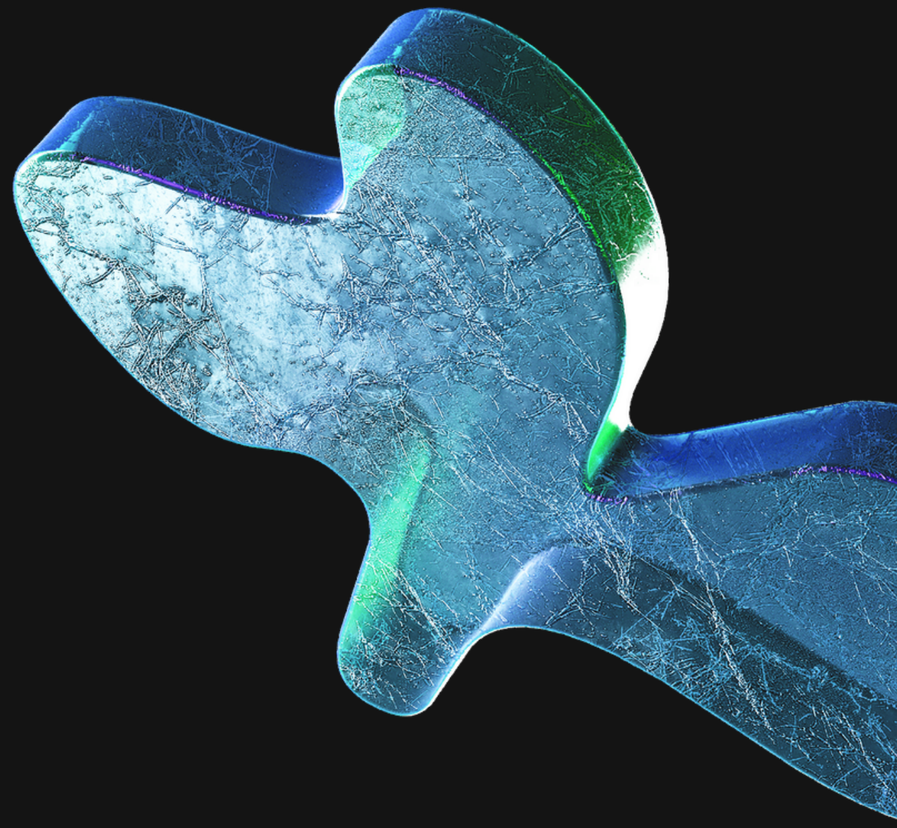
i2i Systems
innovation to integration

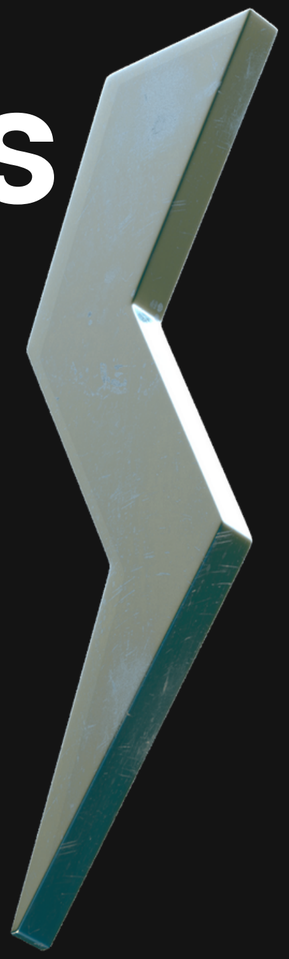
A Pragmatic Approach



Elifnur Kabalcı

July
19

Contains



- The Evils of Duplication
- Orthogonality
- Reversibility
- Tracer Bullets
- Prototypes and Post-it Notes
- Domain Languages
- Estimating

i2i Systems
innovation to integration

Evils of Duplication

Imposed Duplication

Developers feel they have no choice—the environment seems to require duplication.

Inadvertent Duplication

Developers don't realize that they are duplicating information.

Impatient Duplication

Developers get lazy and duplicate because it seems easier.

Interdeveloper Duplication

Multiple people on a team (or on different teams) duplicate a piece of information.

i2i Systems
innovation to integration

Imposed Duplication

- Developers feel they have no choice—the environment seems to require duplication.
- Multiple Representations of Information: In client-server connections, information is provided in the form of templates and implemented as procedures in the used programming languages, which makes the developer feel obligated and prone to duplication.
- Documentation in case: A programmer who creates reporting first thinks that the code will be cleaner, but falling into duplication is very likely. This is because the balance of commands within the code content may not be established with reporting.
- Language Issues: While general-purpose languages have measures against duplication, languages like Object Pascal lack such measures. A function can be defined multiple times in Object Pascal.

i2i Systems
innovation to integration

Inadvertent Duplication

- Developers don't realize that they are duplicating information.
- It arises from creating functions to call ready-to-use methods.

```
class Line {
 public:
  Point  start;
  Point  end;
  double length() { return start.distanceTo(end); }
};
```

i2i Systems
innovation to integration

Impatient Duplication

- Developers get lazy and duplicate because it seems easier.
- Due to time constraints, directly using code from the internet or other sources also leads to duplication.

i2i Systems
innovation to integration

Interdeveloper Duplication

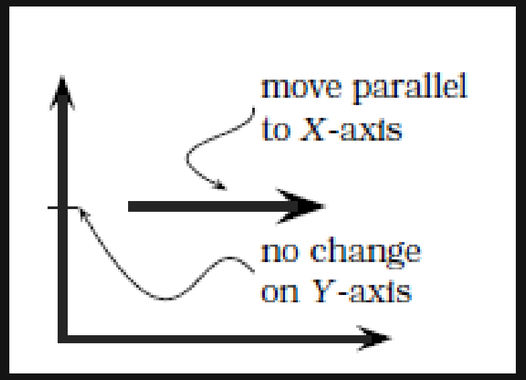
- Multiple people on a team (or on different teams) duplicate a piece of information.
- When multiple modules are worked on within the same project, and different individuals intervene in the same project, duplication can occur.
- Identifying it is the most challenging part.

i2i Systems
innovation to integration

Orthogonality

Increase Productivity

Prepared feasible building blocks are used. This provides efficiency in software development by enabling faster code writing.

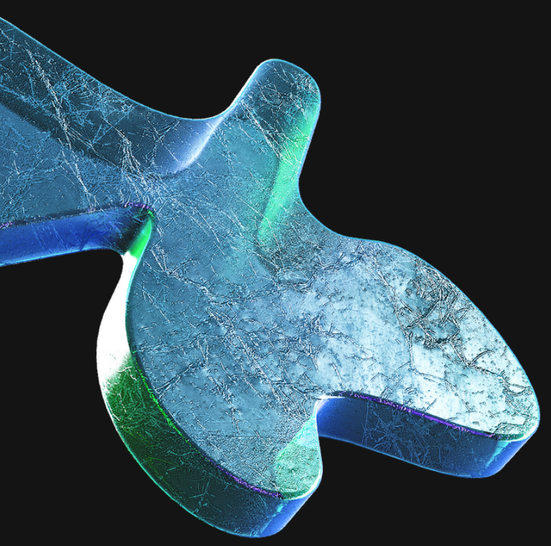


Reduced Risks

Since the code is organized into modules, if an error occurs in one place, it can be resolved without affecting other parts.

i2i Systems
innovation to integration

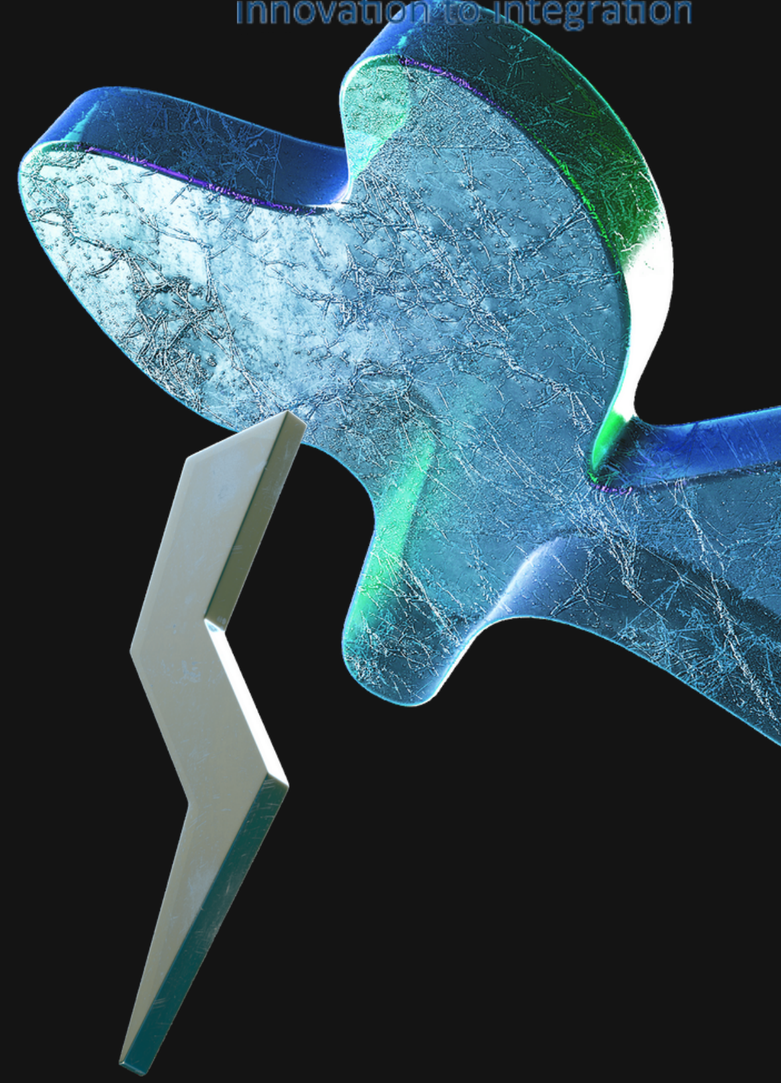
DRY:
Dont Repeat Yourself



i2i Systems
innovation to integration

How We Design Orthogonality?

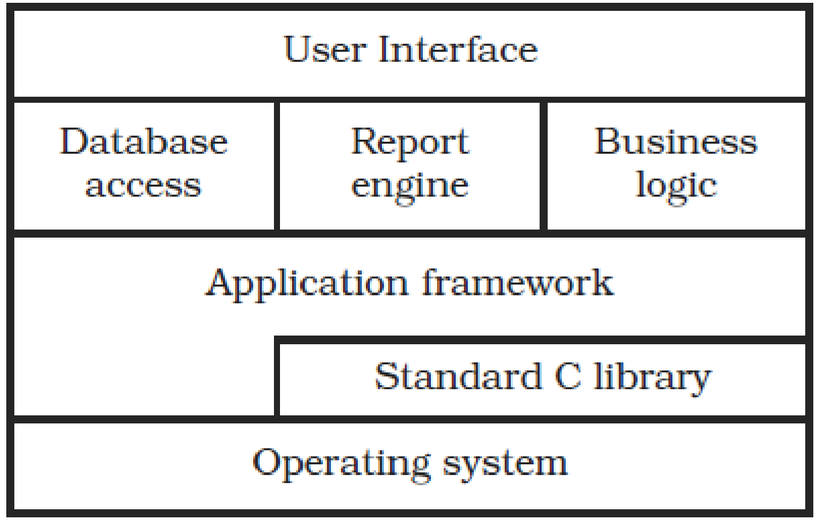
Project Team
Design
    Modules, Component-based, Layered
Toolkits and Libraries
Coding
    Avoiding global data
Testing
Documentation



i2i Systems
innovation to integration

Layered Design Table

Figure 2.1. Typical layer diagram



i2i Systems
innovation to integration

Reversibility

- Flexible Architecture:
  - Architecture
  - Deployment
  - Vendor Integration

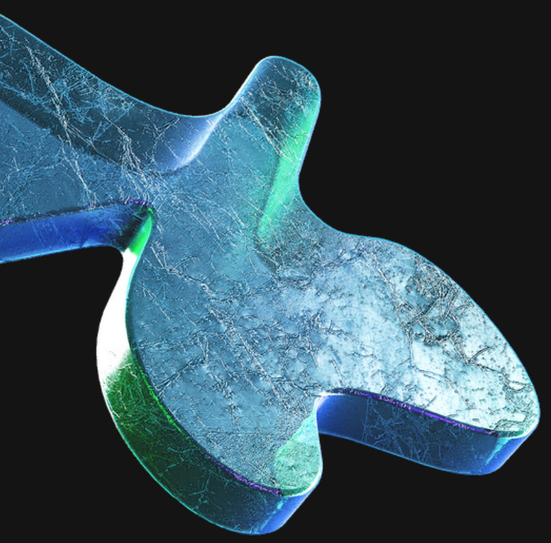
i2i Systems
innovation to integration

Tracer Bullet

- Advantages:
- Users get to see something working early
- Developers build a structure to work in
- You have an integration platform
- You have something to demonstrate
- You have a better feel for progress

i2i Systems
innovation to integration

Prototypes and Post-it Notes



i2i Systems
innovation to integration

You can prototype:

- Architecture
- New functionality in an existing system
- Structure or contents of external data
- Third-party tools or components
- Performance issues
- User interface design

i2i Systems
innovation to integration

How to use Prototypes:

Correctness

You may be able to use dummy data where appropriate.

Completeness

The prototype may function only in a very limited
sense, perhaps with only one preselected piece of input data and
one menu item.

Robustness

Error checking is likely to be incomplete or missing entirely. If you stray from the predefined path, the prototype may
crash and burn in a glorious display of pyrotechnics. That's okay.

Style

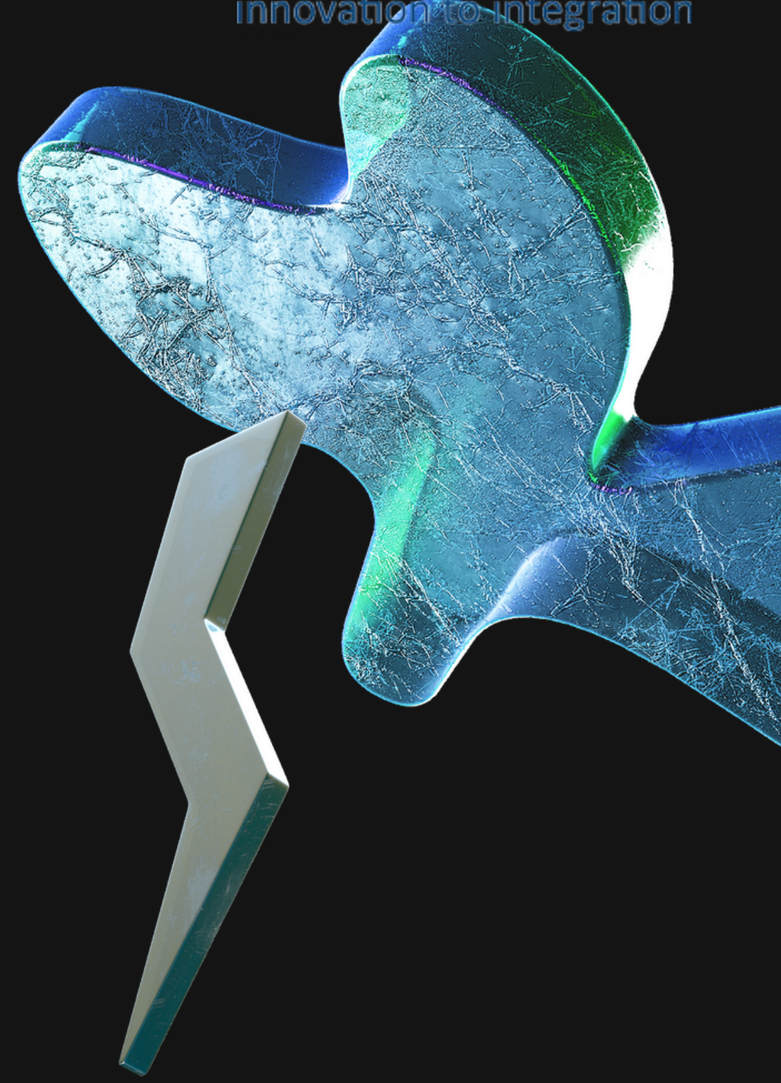
It is painful to admit this in print, but prototype code probably
doesn't have much in the way of comments or documentation.
You may produce reams of documentation as a result of your experience with the prototype, but comparatively very little on the prototype system itself.

i2i Systems
innovation to integration

Domain Languages

The limits of language are the limits of one's world.

Ludwig Wittgenstein



i2i Systems
innovation to integration

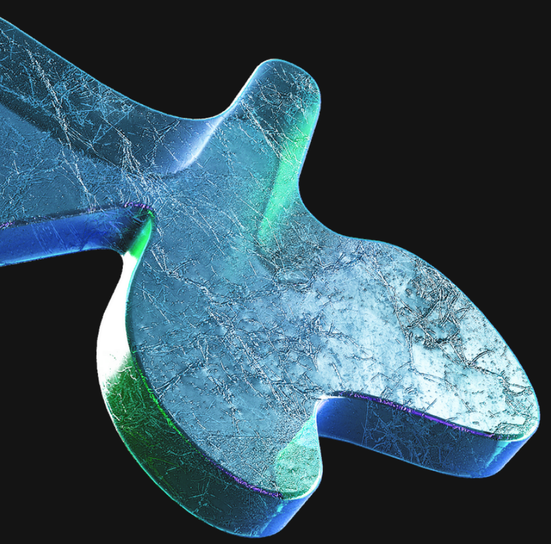
Implementing a Mini-Language

Data Languages

Data languages are languages designed for data processing and manipulation. These languages are used for tasks such as working with databases, data analysis, querying, and transformation. They typically focus on data-oriented operations and tasks. Data languages are effective in handling structured data, querying and processing it. Examples include SQL (Structured Query Language), NoSQL query languages (e.g., MongoDB query language), and R.

Imperative Languages

Imperative languages use a sequence of commands to specify the flow of a program and its steps. These languages provide detailed instructions on how the program should be executed and allow the programmer to specify what should be done with instructions. They involve operations like variable assignment, loops, and conditional statements. Commonly used general-purpose programming languages like C, C++, Java, and Python are imperative languages.



Domain-Specific Errors

If you are writing in the problem domain, you can also perform domain-specific validation, reporting problems in terms your users can understand. Take our switching application on on the facing page. Suppose the user misspelled the format name:

```
From X25LINE1 (Format=AB123)
```

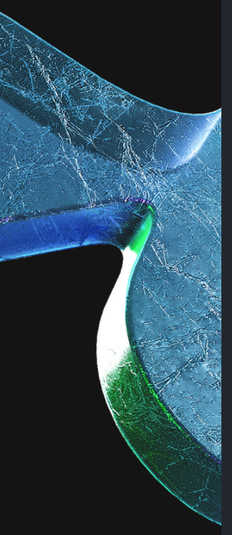
If this happened in a standard, general-purpose programming language, you might receive a standard, general-purpose error message:

```
Syntax error: undeclared identifier
```

But with a mini-language, you would instead be able to issue an error message using the vocabulary of the domain:

```
"AB123" is not a format. Known formats are ABC123,
        XYZ43B, PDQB, and 42.
```

Yacc Example



```
OP_PLUS             "+"
OP_MINUS            "-"
OP_DIV              "/"
OP_MULT             "*"
OP_OP               "("
OP_CP               ")"
OP_DBLMULT          "**"
OP_OC               "“"
OP_CC               "”"
OP_COMMA            ","

/*** Keywords ***/

KW_AND      "and"
KW_OR       "or"
KW_NOT      "not"
KW_EQUAL    "equal"
KW_LESS     "less"
KW_NIL      "nil"
KW_LIST     "list"
KW_APPEND   "append"
KW_CONCAT   "concat"
KW_SET      "set"
KW_DEFFUN   "deffun"
KW_FOR      "for"
KW_IF       "if"
KW_EXIT     "exit"
KW_LOAD     "load"
KW_DISP     "disp"
KW_TRUE     "true"
KW_FALSE    "false"
```

```
VALUE                   [0-9]|[1-9][0-9]*
ID                              [a-zA-Z][a-zA-Z0-9]*
float           [1-9]+"f"+[1:9]*
STRING                  ["](.*?)["]
NEWLINE                 "n"
COMMENT                 ";;".*


/*** Printing ***/

%%
(\r\n|\r|\n) {return 0;}

{OP_PLUS}               {printf("OP_PLUS\n");}
{OP_MINUS}              {printf("OP_MINUS\n");}
{OP_DIV}                {printf("OP_DIV\n");}
{OP_MULT}               {printf("OP_MULT\n");}
{OP_CP}                 {printf("OP_CP\n");}
{OP_OP}                 {printf("OP_OP\n");}
{OP_DBLMULT}    {printf("OP_DBLMULT\n");}
{OP_OC}                 {printf("OP_OC\n");}
{OP_CC}                 {printf("OP_CC\n");}
{OP_COMMA}              {printf("OP_COMMA\n");}
```

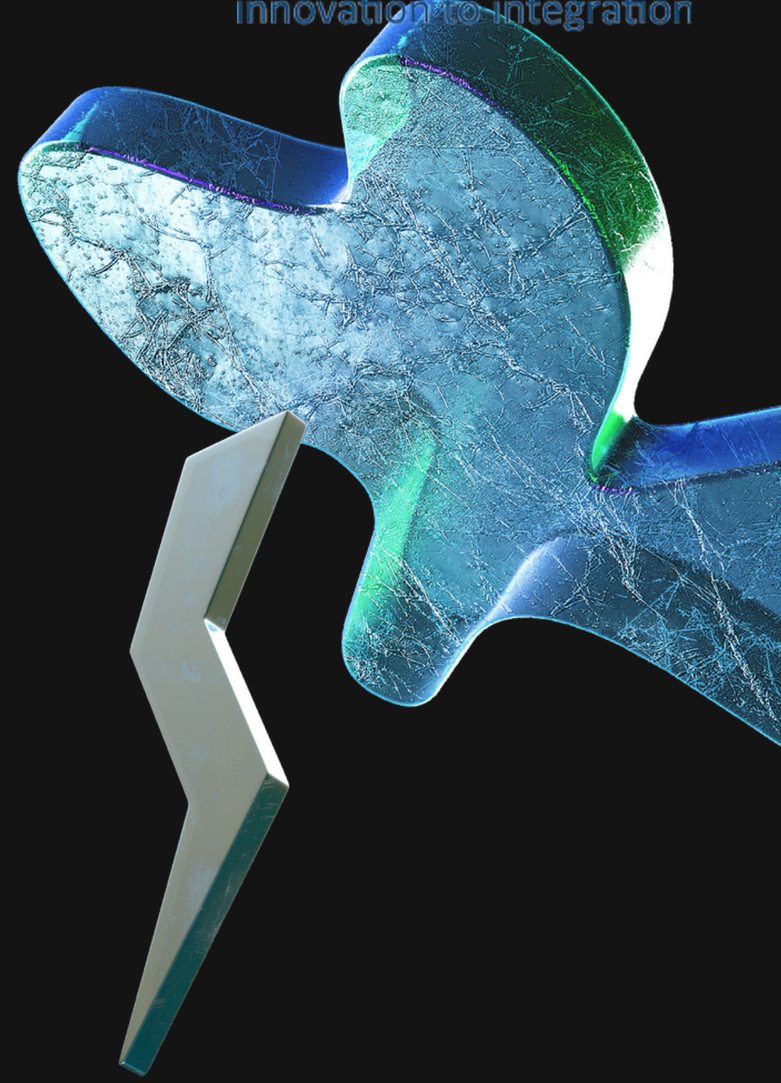
```
[[1-9][:digit:]]+       {printf("VALUE\n");}
{VALUE}                         {printf("VALUE\n");}
[[:alpha:]][[:alnum:]]* {printf("IDENTIFIER\n");}
{STRING}                        {printf("STRING\n");}
[[:space:]]+                /* ignore space */
{COMMENT}                       {printf("COMMENT\n");}
{NEWLINE}               {++numberofline}
{FLOAT}                 {printf("FLOAT\n");}
/*** Errors ***/

([0-9][0-9a-zA-Z]+)|. {
    printf("SYNTAX_ERROR at %d. line, %s cannot be tokenized.\n", numberofline, yytext);
    exit(EXIT_FAILURE);
}
```

i2i Systems
innovation to integration

Stand-Alone and Embedded Languages

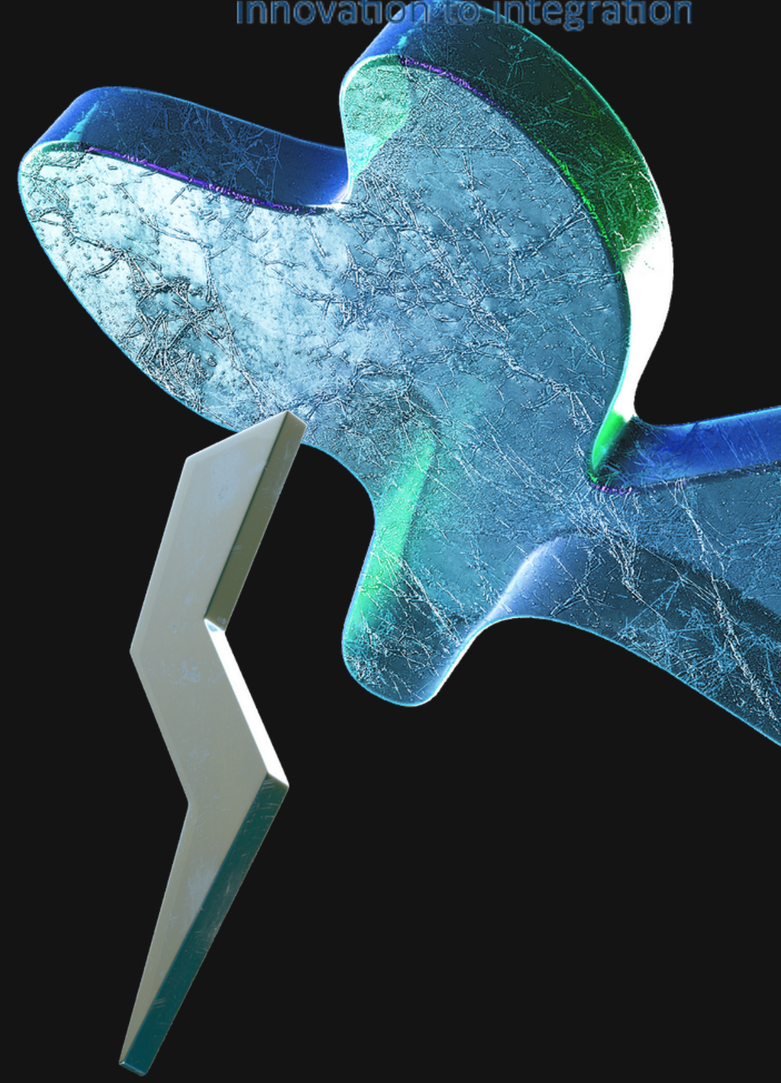
Embedded languages are referred to as languages created for a specific purpose, while stand-alone languages are designed for general purposes.



i2i Systems
innovation to integration

Easy Development or Easy Maintenance?

Easy development is effective for short-term, fast-deadline projects, while easy maintenance is effective for long-term, efficiency-focused projects.



i2i Systems
innovation to integration

Estimating

- How Accurate Is Accurate Enough?
- Where Do Estimates Come From?
- Understand What's Being Asked
- Build a Model of the System
- Break the Model into Components
- Give Each Parameter a Value
- Calculate the Answers

i2i Systems
innovation to integration

Estimating Project Schedules

- Check requirements
- Analyze risk
- Design, implement, integrate
- Validate with the users

i2i Systems
innovation to integration

Sources

The Pragmatic Programmer-First Edition
chat.openai.com
https://github.com/elifnurkabalci

i2i Systems
innovation to integration

Do you have any questions?

Thank You