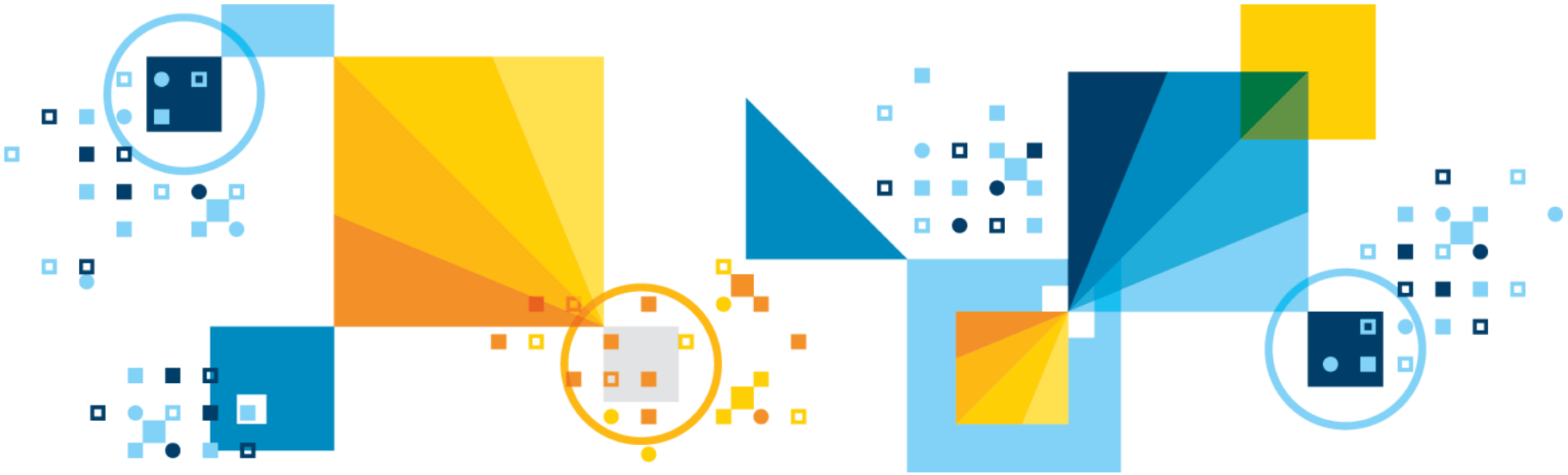
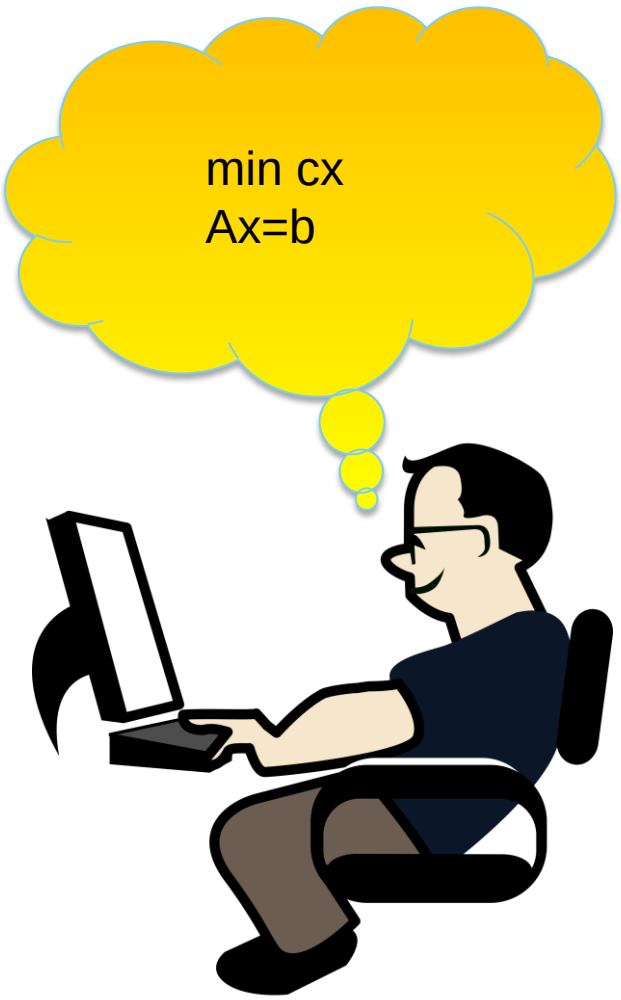
CPLEX APIs

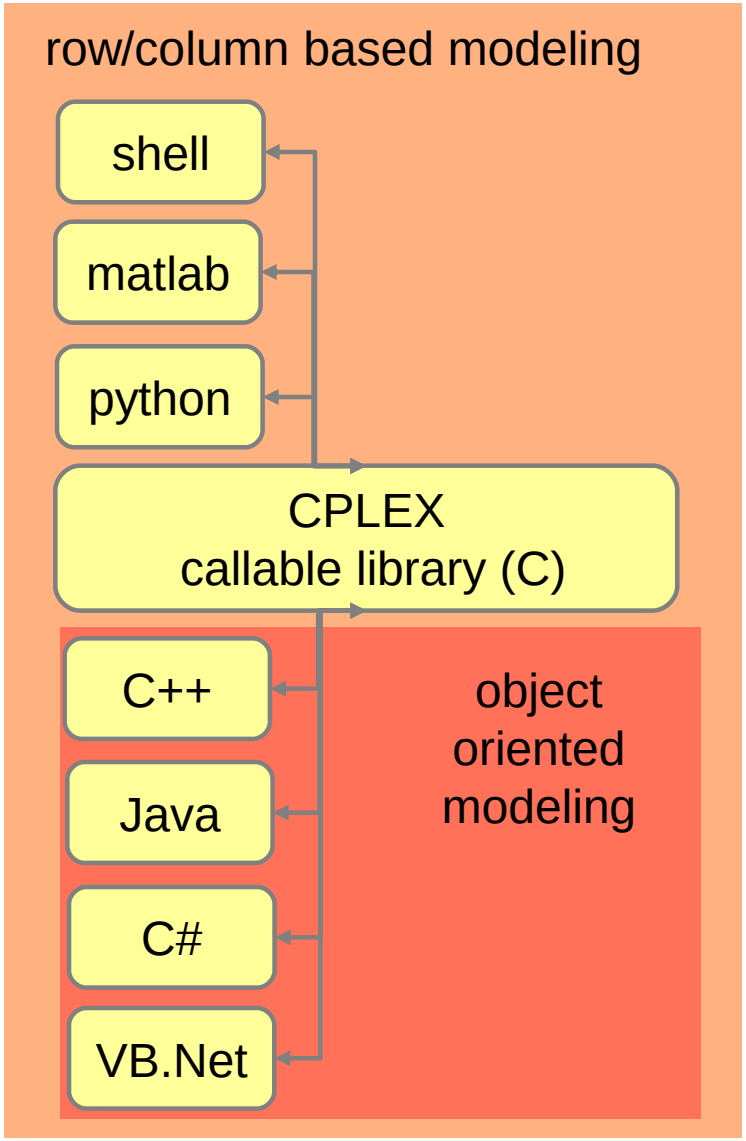
@CPLEX School 2017 Montreal



© 2017 IBM Corporation

Interfacing with CPLEX: APIs





© 2017 IBM Corporation

Interfacing with CPLEX: modeling languages

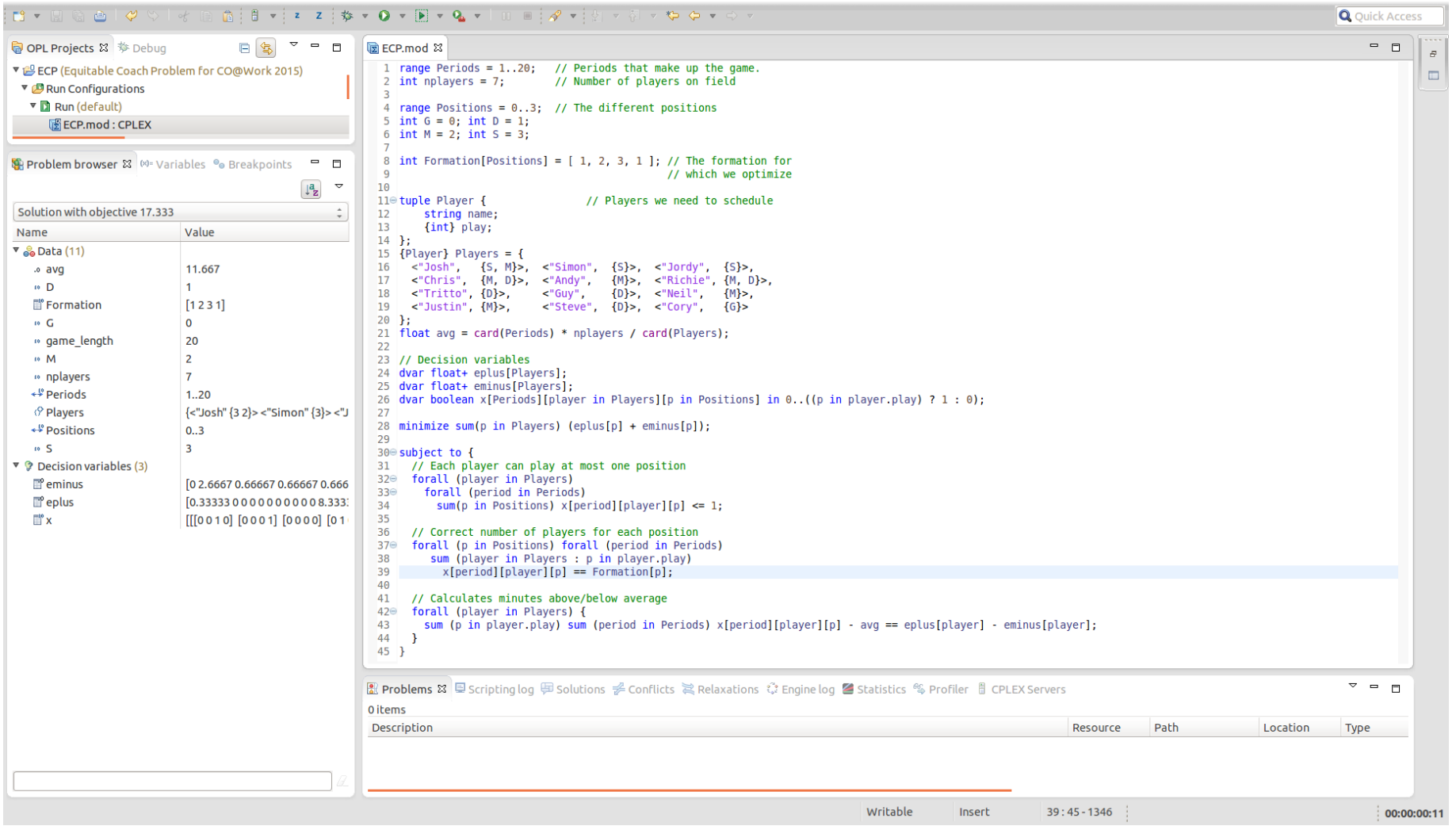
Optimization Programming Language (OPL)

- write models in a more descriptive form
- write models in a more compact form
- faster prototyping, easier maintenance
- easier access to data (Excel, database, ...)

```
forall (i in I)
  sum (j in J)
    sum (k in K)
      x[i][j][k] <= 1;
```

- scriptable
- model editor
- IDE support (eclipse based)

© 2017 IBM Corporation



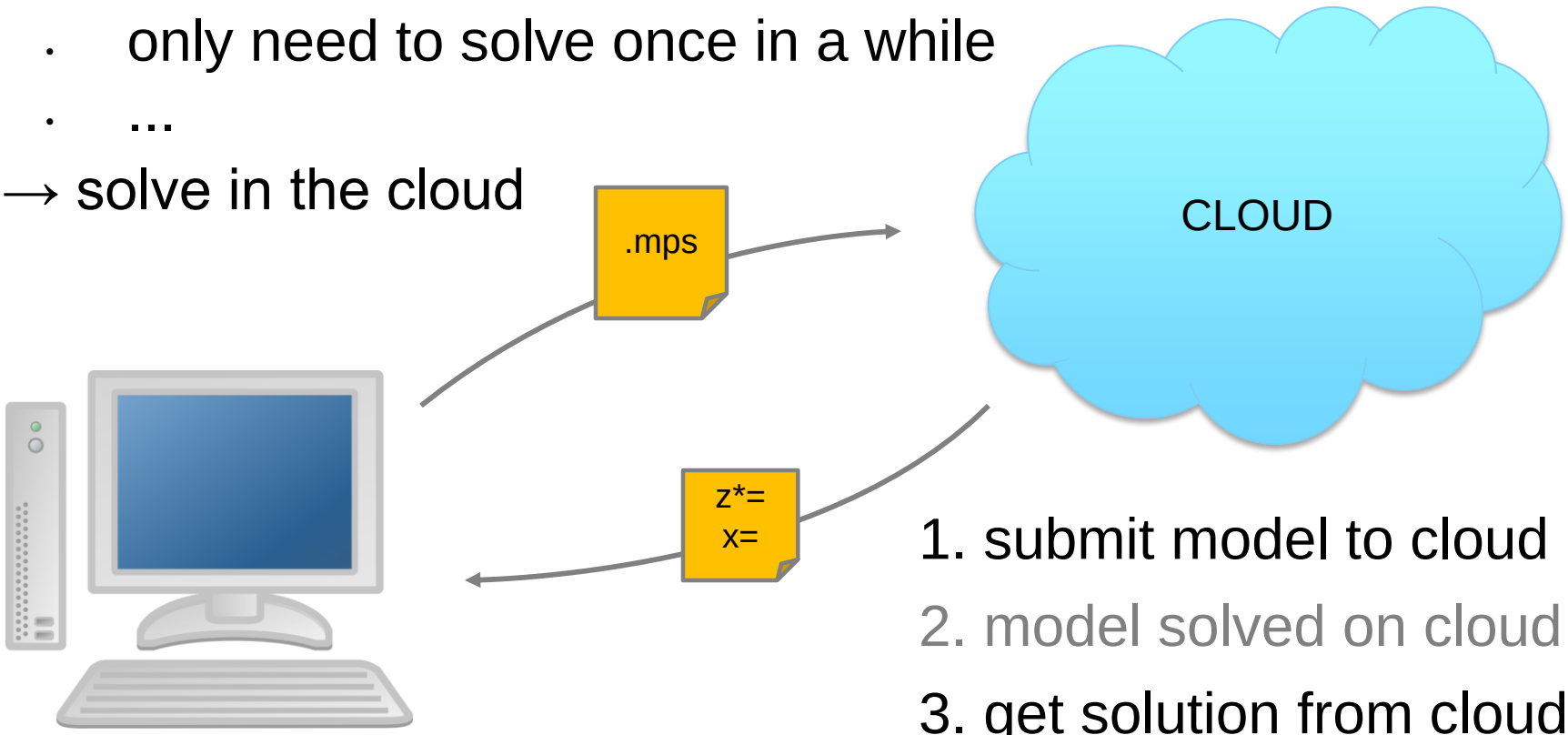
© 2017 IBM Corporation

CPLEX in the CLOUD

What if no local resources to solve model?

- model too hard
- only need to solve once in a while
- ...

→ solve in the cloud



1. submit model to cloud
2. model solved on cloud
3. get solution from cloud

© 2017 IBM Corporation

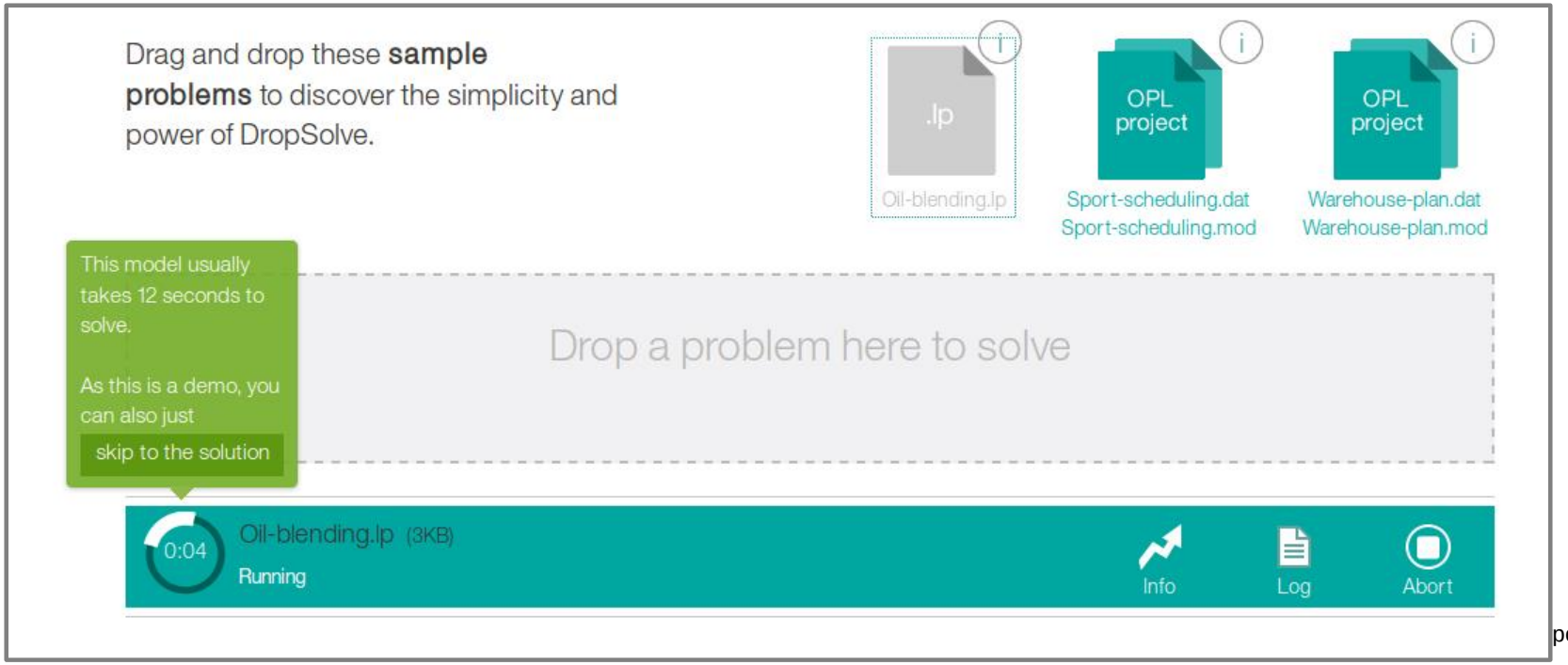
CPLEX in the CLOUD

Two ways to access CPLEX in the cloud

1. DropSolve

https://dropsolve-oaas.docloud.ibmcloud.com/software/analytics/docloud

Drag and drop your model file from disk to web browser



poration

CPLEX in the CLOUD

Two ways to access CPLEX in the cloud

2. REST API

https://developer.ibm.com/docloud/
https://developer.ibm.com/docloud/docs/welcome/

Access the solve service via its REST API

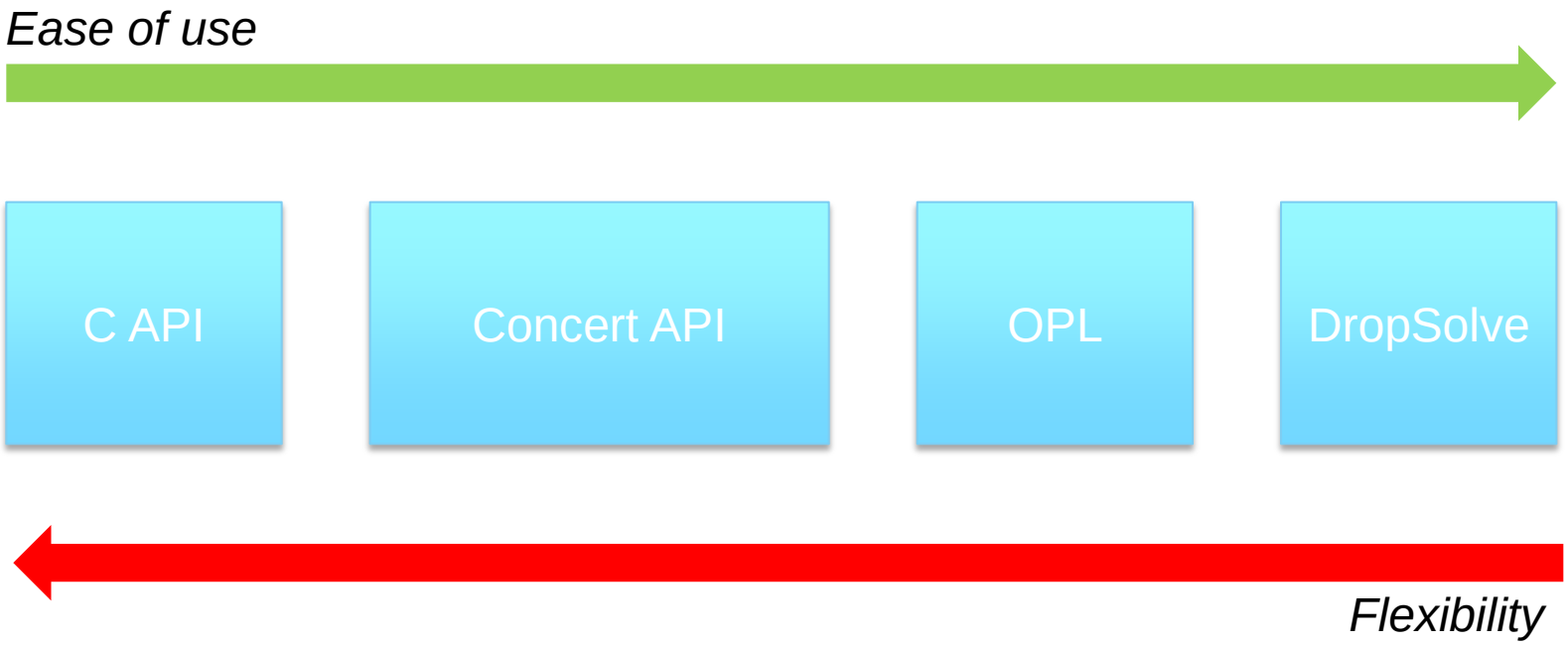
- ready-to-use clients provided for Java and Python, e.g.

```
JobExecutor executor = JobExecutorFactory.createDefault();
JobClient jobclient = JobClientFactory.createDefault(BASE_URL,
                                                     APIKEY_CLIENTID);
jobclient.newRequest().input(new File("model.mps"))
                      .output(new File("x.sol"))
                      .execute(executor).get();
```

- With any HTTP client (cURL, ...)

© 2017 IBM Corporation

So what should you use?



- Hard to tell in general
- Depends on use case
- But…

© 2017 IBM Corporation

docplex *https://pypi.python.org/pypi/docplex*

- Pure Python modeling API
- Opensource (pypi, github)
- Support local and cloud solves
- Why Python?
  - You don’t need to learn yet another DSL
  - Python provides enough *syntactic sugar* for modeling
  - Get direct access to the whole Python software ecosystem (quite strong in the scientific community: numpy, scipy, iPython/Jupyter notebooks, etc…)

© 2017 IBM Corporation

Callable library: CPLEX C API

- Used for tight/efficient integration into applications
- C libraries are broadly callable also from higher level languages

  (no need to stick to C for using the C API!)

- Most comprehensive CPLEX API:

  if it is not in the C API, it doesn't exists.
- Also the harder to use (C itself is rather low-level)

© 2017 IBM Corporation

Callable library 101

Based on two opaque objects:

- **environment**: parameters/callbacks/license checking
- **problem**: problem data (variables, constraints, objective…)

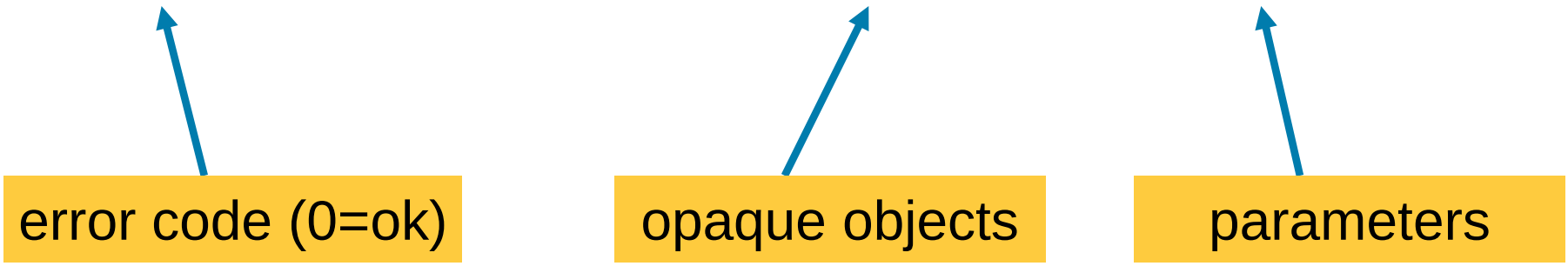
CPX(C)ENVptr	CPXopenCPLEX/CPXcloseCPLEX
CPX(C)LPptr	CPXcreateprob/CPXfreeprob

© 2017 IBM Corporation

Callable library 101

- The two objects can be inspected/modified by using appropriate functions
- Almost all functions follow the same pattern:

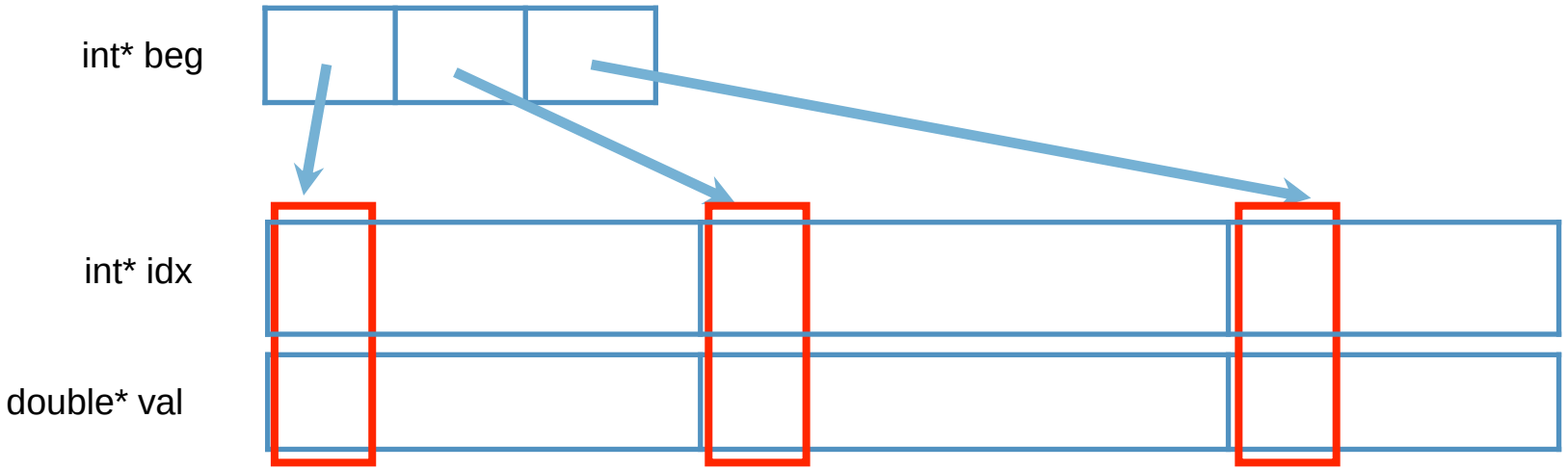
```
int CPXXfunction(environment[,problem],...);
```



© 2017 IBM Corporation

Callable library 101

- As with many C libraries, only basic data types are supported:
  - Scalars (by value and/or pointer): double/int
  - Arrays: double*/int*
  - Strings: char*
- For passing around matrices, a sparse data structure is used:



© 2017 IBM Corporation

Help/Documentation

IBM Knowledge Center: online documentation

https://www.ibm.com/support/knowledgecenter/SSSA5P

IBM developerWorks: community forum

https://www.ibm.com/developerworks

© 2017 IBM Corporation

Legal Disclaimer

- © IBM Corporation 2017. All Rights Reserved.
- The information contained in this publication is provided for informational purposes only. While efforts were made to verify the completeness and accuracy of the information contained in this publication, it is provided AS IS without warranty of any kind, express or implied. In addition, this information is based on IBM's current product plans and strategy, which are subject to change by IBM without notice. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this publication or any other materials. Nothing contained in this publication is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.
- References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in this presentation may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.
- Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.
- Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.
- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- Other company, product, or service names may be trademarks or service marks of others.

© 2017 IBM Corporation