CP-2010

CP Standardization Discussion

Jacob Feldman
www.cpstandards.org
www.cpstandard.wordpress.com
j.feldman@4c.ucc.ie

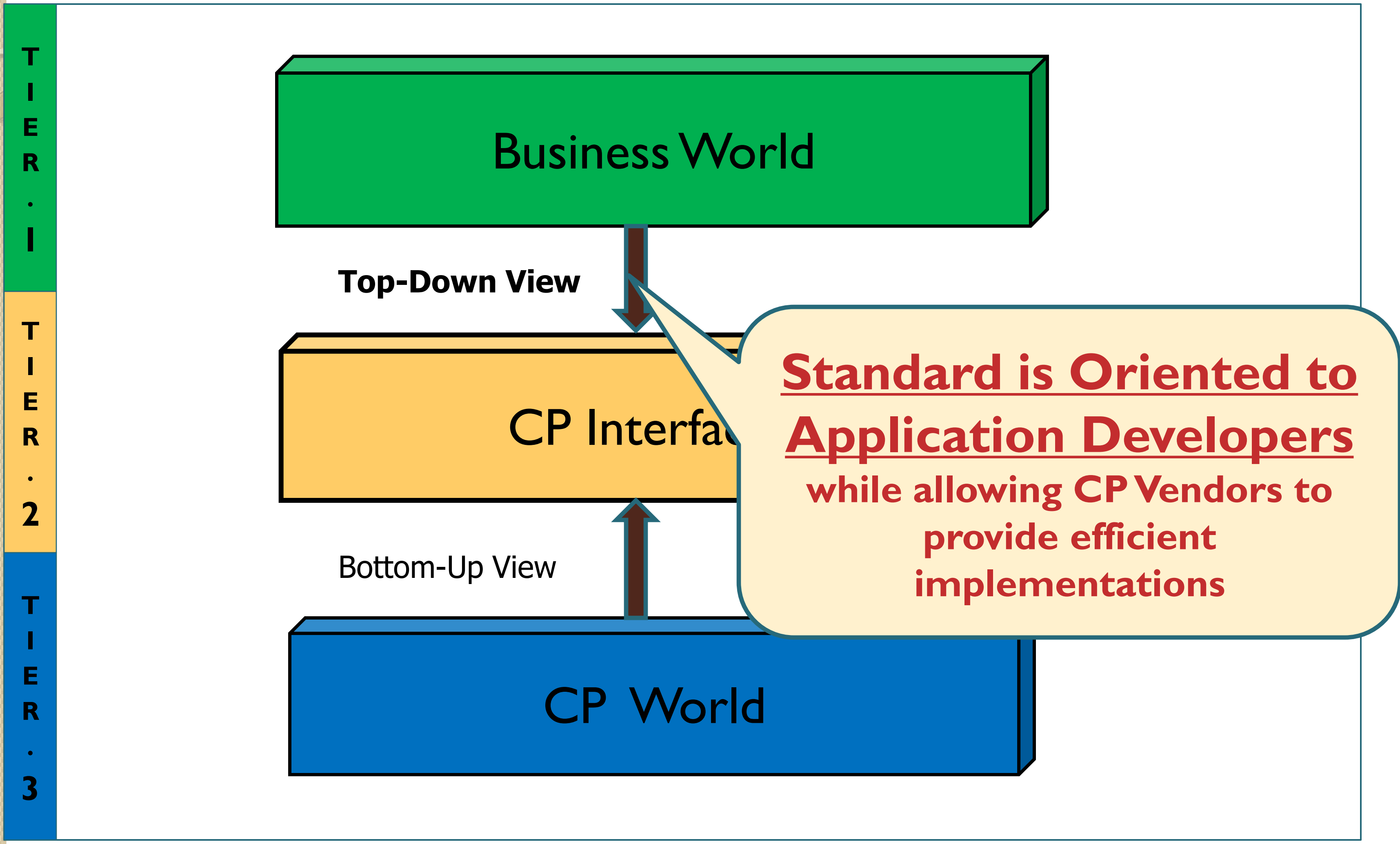
September 10, 2010

Key Standardization Objectives

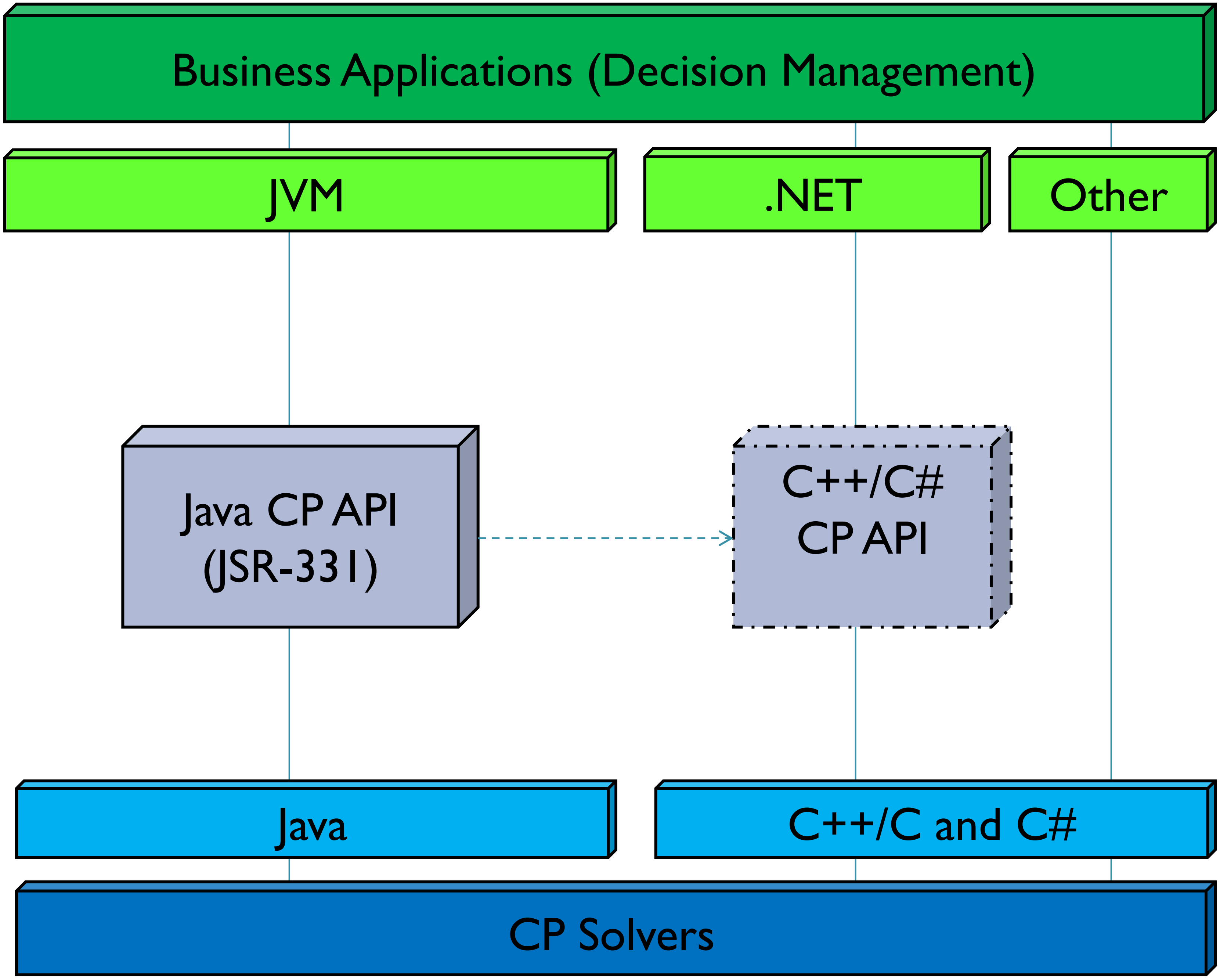
- Make CP more accessible for business software developers
- Provide a vendor-neutral standard CP interface(s)
  - Ability to switch between underlying CP solvers without changing application code
- Assist CP vendors in creating practical and efficient standard implementations
- Not to limit innovation

www.cpstandards.org

CP Standardization Perspective



Standard is Oriented to Application Developers while allowing CP Vendors to provide efficient implementations


Business Applications (Decision Management)
JVM
.NET
Other
Java CP API (JSR-331)
C++/C# CP API
Java
C++/C and C#
CP Solvers

We started "small"

- Start with only commonly used concepts for CSP Definition and Resolution
- But standard should be extensive enough to solve real-world problems
- Ability to expand later on:
  - Interfaces for Scheduling, Configuration, Routing
  - Interfaces to other solvers (LP, MIP, hybrids, ..)
  - Advanced techniques: explanations, reformulations, visualization, …
- See a roadmap at cpstandard.wordpress,com

JSR-331 – Java Specification Request

- Java Constraint Programming API under the roof of the Java Community Process (JCP) www.jcp.org
- JSR-331 covers key concepts and makes important design decisions related to the Standard Representation and Resolution of constraint satisfaction and optimization problems
- JSR-331Early Draft (a new iteration 0.6.1) is available for public review at www.cpstandards.org

Great Community Input

- Heated Arguments and Constructive Contributions
- Special thanks to:
  - CP vendors:
    - Gecode, Choco, IBM/ILOG, G12, JaCoP, 4C
  - CP experts:
    - Helmut S., Peter S., Nicolas B., and many others
  - Java experts for the JSR-331 Expert Group

Only six major concepts

- Problem
  - ConstrainedVariable
  - Constraint
- Solver
  - SearchStrategy
  - Solution

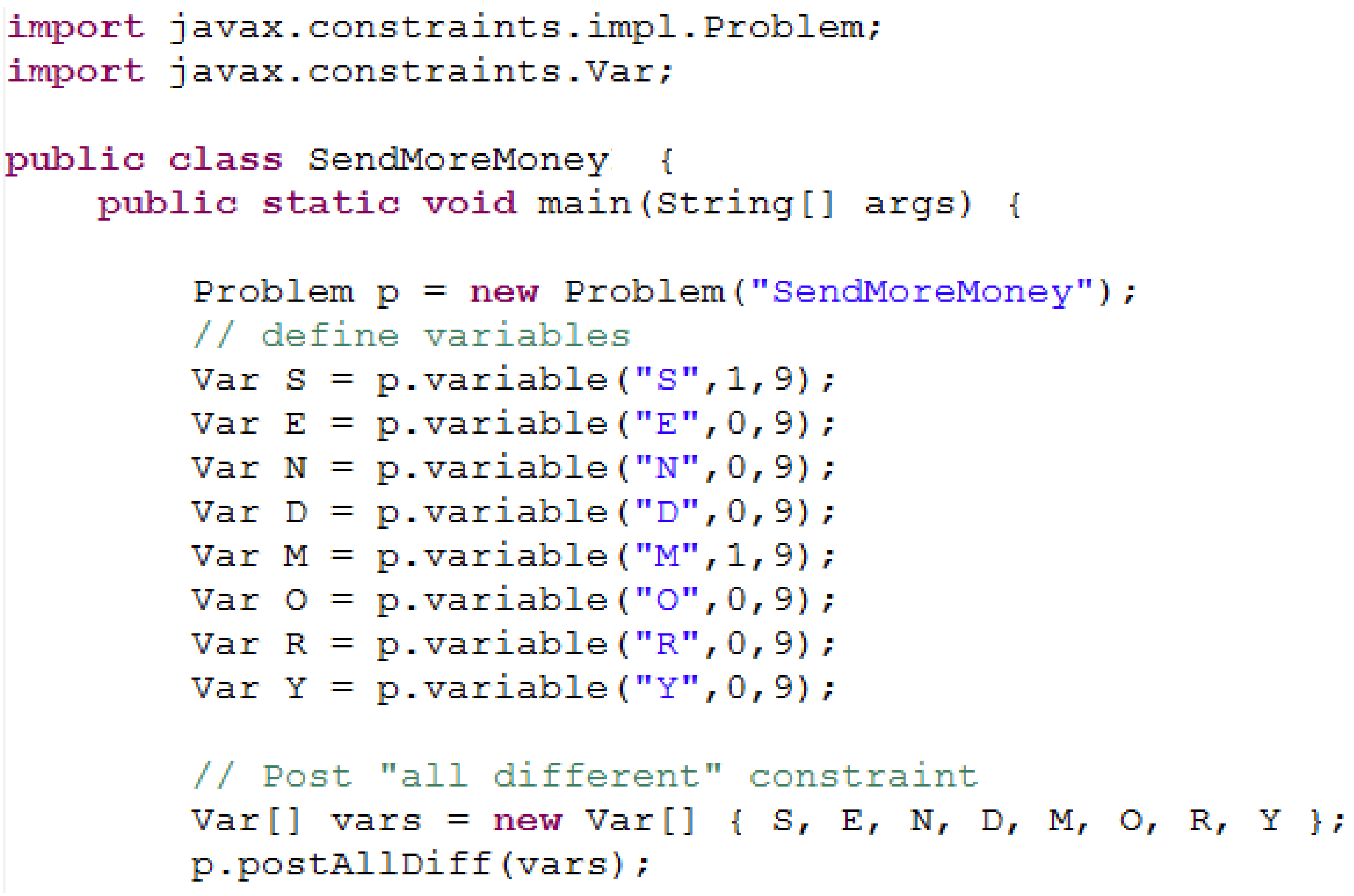
Examples

```
Problem p = new Problem();
Var x = p.variable("X", 0, 10);
Var y = p.variable("Y", 0, 10, DomainType.DOMAIN_SPARSE);
p.post(x,"<",y);
p.post(x.plus(y),"=",z);
p.post(values,vars,"<", 16);
p.postAllDiff(vars);
p.postElement(var, indexVar, ">=", value);
…
Solution solution = p.getSolver().findSolution();
Solution.log();
```

Constraints

- Currently Included:
  - All Basic
  - Linear
  - AllDifferent
  - Element
  - Cardinality
  - GlobalCardinality
- Under Consideration:
  - “regular”,“diffn”, “cumulative”, …

SEND+MORE=MONEY in Java (1)



```
import javax.constraints.impl.Problem;
import javax.constraints.Var;

public class SendMoreMoney  {
    public static void main(String[] args) {

        Problem p = new Problem("SendMoreMoney");
        // define variables
        Var S = p.variable("S",1,9);
        Var E = p.variable("E",0,9);
        Var N = p.variable("N",0,9);
        Var D = p.variable("D",0,9);
        Var M = p.variable("M",1,9);
        Var O = p.variable("O",0,9);
        Var R = p.variable("R",0,9);
        Var Y = p.variable("Y",0,9);

        // Post "all different" constraint
        Var[] vars = new Var[] { S, E, N, D, M, O, R, Y };
        p.postAllDiff(vars);
```

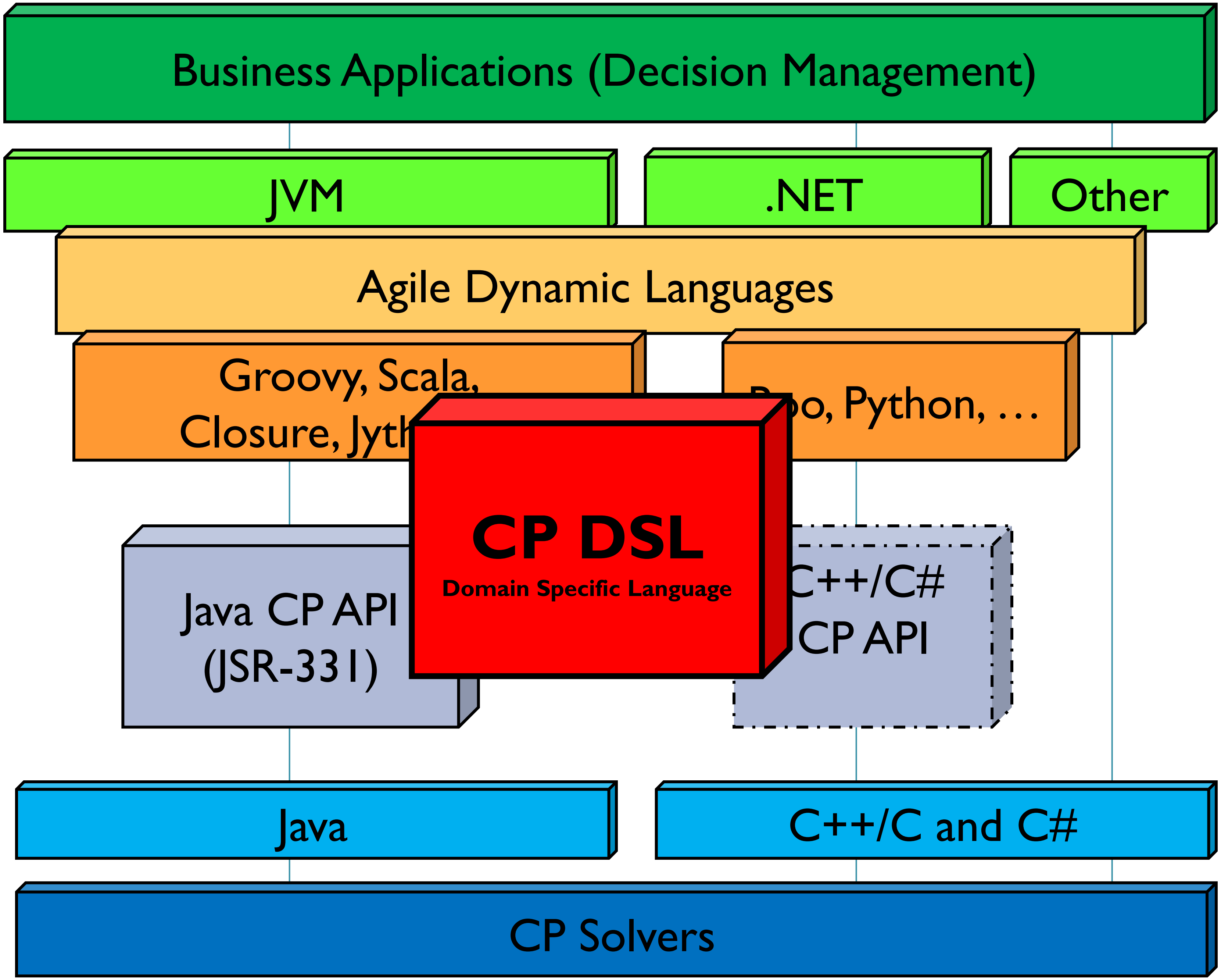
SEND+MORE=MONEY in Java (2)

```
        // Define expression SEND
        int coef1[] = { 1000, 100, 10, 1 };
        Var[] sendVars = { S, E, N, D };
        Var SEND = p.scalProd(coef1, sendVars);
        SEND.setName("SEND");
        // Define expression MORE
        Var[] moreVars = { M, O, R, E };
        Var MORE = p.scalProd(coef1, moreVars);
        MORE.setName("MORE");
        // Define expression MONEY
        Var[] moneyVars = { M, O, N, E, Y };
        int coef2[] = { 10000, 1000, 100, 10, 1 };
        Var MONEY = p.scalProd(coef2, moneyVars);
        MONEY.setName("MONEY");

        // Post constraint SEND + MORE = MONEY
        p.post(SEND.plus(MORE),"=",MONEY);

        // Problem Resolution
        p.getSolver().findSolution();
        p.log("Solution: " + SEND + " + " + MORE + " = " + MONEY);
    }
}
```

Moving From Java To…


Business Applications (Decision Management)
JVM
.NET
Other
Agile Dynamic Languages
Groovy, Scala,
Closure, Jyth
o, Python, …
CP DSL
Domain Specific Language
Java CP API
(JSR-331)
C++/C#
CP API
Java
C++/C and C#
CP Solvers

SEND+MORE=MONEY in Groovy

```
import javax.constraints.groovy.ProblemGroovy;

ProblemGroovy p = new ProblemGroovy("SendMoreMoney");
// define variables
S = p.variable("S",1..9)
E = p.variable("E",[0,1,2,3,4,5,6,7,8,9])
N = p.variable("N",0,9)
D = p.variable("D",0,9)
M = p.variable("M",1,9)
O = p.variable("O",0,9)
R = p.variable("R",0,9)
Y = p.variable("Y",0,9)

p.postAllDifferent([ S, E, N, D, M, O, R, Y ])

// Post constraint SEND + MORE = MONEY
SEND = 1000*S + 100*E + 10*N + D
MORE = 1000*M + 100*O + 10*R + E
MONEY = 10000*M + 1000*O + 100*N + 10*E + Y
p.post(SEND + MORE, "=", MONEY)

// Problem Resolution
p.solver.findSolution()
p.log "Solution: ${SEND} + ${MORE} = ${MONEY}"
```

SEND+MORE=MONEY in CP DSL (?)

```
// define variables
S = variable("S",1..9)
E = variable("E",0..9)
N = variable("N",0,9)
D = variable("D",0,9)
M = variable("M",1,9)
O = variable("O",0,9)
R = variable("R",0,9)
Y = variable("Y",0,9)

postAllDifferent([ S, E, N, D, M, O, R, Y ])

// Post constraint SEND + MORE = MONEY
SEND = 1000*S + 100*E + 10*N + D
MORE = 1000*M + 100*O + 10*R + E
MONEY = 10000*M + 1000*O + 100*N + 10*E + Y
post(SEND + MORE, "=", MONEY)

// Problem Resolution
findSolution()
log "Solution: ${SEND} + ${MORE} = ${MONEY}"
```

JSR-331 – Java Specification Request

- JSR-331Early Draft will be finalized soon
- Provide comments at www.cpstandards.org and contribute to creation of the Technology Compatibility Kit (TCK)
- There are 3 working JSR-331 implementations: more are welcome. Send download requests to j.feldman@4c.ucc.ie
- Finally, JSR-331 has been nominated by JCP to the most innovative JSR of 2010. It is an honor, but:
  - It tells more about unfamiliarity of Java world with CP
  - It will contribute to CP recognition among Java developers

Discussion Panel

- Peter Stucky (G12)
- Laurent Michel (Comet)
- Chris Jefferson (Minion)
- Nicolas Beldiceanu (Choco)
- Radek Szymanek (JaCoP)
- Helmut Simonis

Discussion Topics

- CP API for main-stream languages
- JSR-331
- CP DSL
- CP XML
- Standard Test Problems
- Integration with MIP/LP
- Integration with Rule Engines, Office,…
- Vertical Add-Ons
- Others

www.cpstandards.org