



CP-2009

CP Standardization Discussion

Jacob Feldman, Narendra Jussien

September 23, 2009

www.cpstandards.org

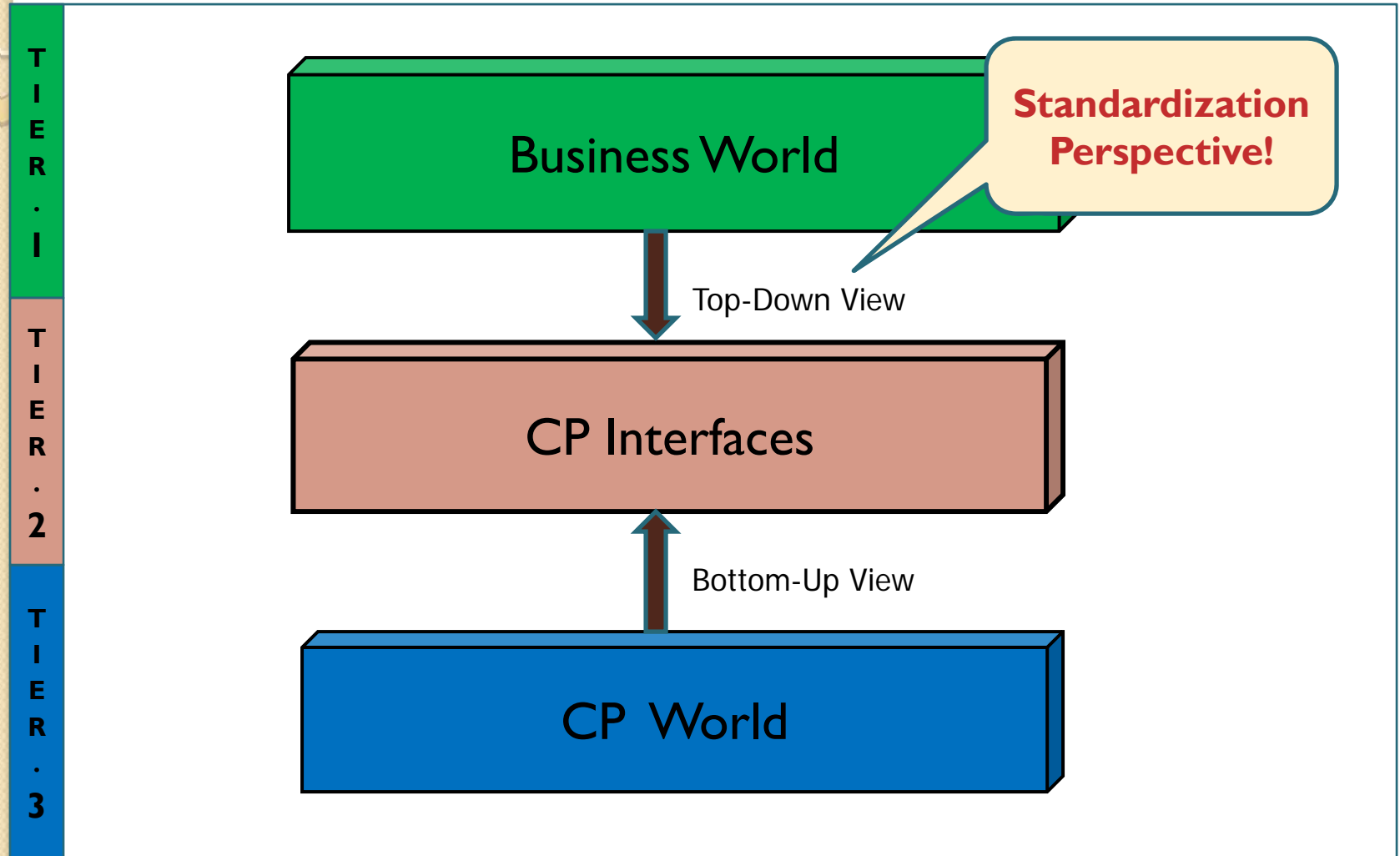
Outline

- Why to Standardize
- What to Standardize
 - Start Small
 - Future Standard Additions
- What Not to Standardize
- How to Standardize
- Current Standardization Status
 - www.cpstandards.org
 - JSR 331 - www.jcp.org

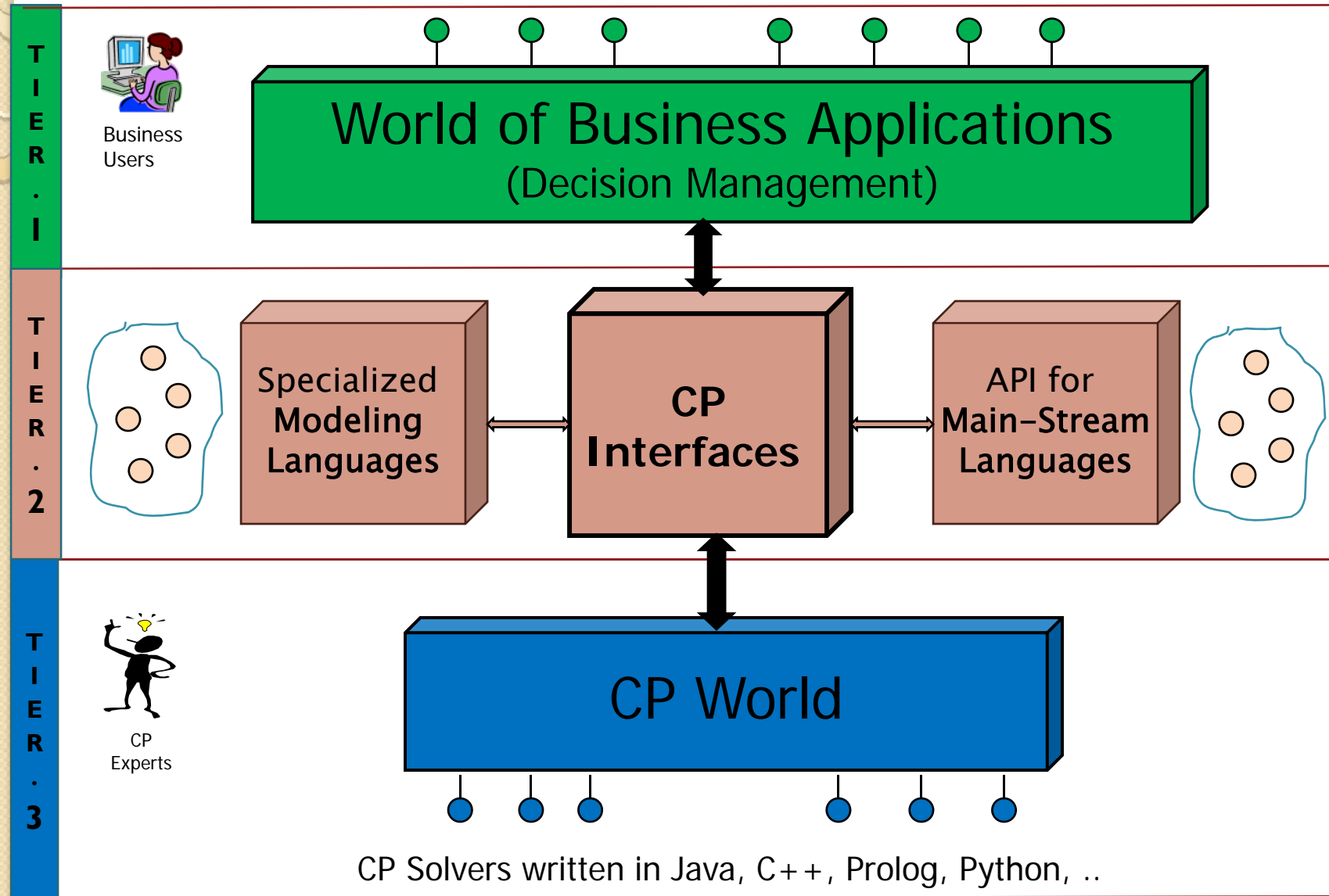
Why to Standardize

- **Just a Quote** (from Nethercote, Stuckey, Becket, Brand, Duck, Tack)
 - Most CP solvers have their own modeling language. This makes it difficult for modelers to experiment with different solvers for a problem
 - A standard language for modeling CP problems will encourage experimentation with and comparisons between different solvers
 - No standard CP modeling language will be perfect - we believe simplicity, expressiveness, and ease of implementation make it a practical choice for a standard language.
- **Utilize existing experience**

CP World and Business Applications



CP-to-Business Interfaces



Key Standardization Objectives

- Make CP more accessible for business software developers
- Provide a vendor-neutral unified CP interface(s)
 - Ability to switch between CP solvers without changing code
- Corresponding Libraries of constraints, search algorithms, CSP problems, and tools supported by the CP community
- Reference Implementations using existing CP solvers

What To Standardize

- Start “Small”:
 - Only commonly used concepts for CSP Definition and Resolution
 - BUT: Extensive enough to solve real-world problems
 - Ability to add user-defined constraints and search strategies
- Ability to expand later on:
 - Interfaces for Scheduling, Configuration, Routing,...
 - Interfaces to other solvers (LP, MIP, hybrids, ..)
 - Advanced techniques: explanations, reformulations, visualization, ...

How To Standardize

- Major CP concepts are de-facto standardized
- Externalize commonly used features from the most popular CP products
- Start with WHAT then define HOW
- Not one “ultimate” language but rather the same standardized concepts presented in different languages
- Common Glossary and Naming Convention

Initial Standardization Targets:

- Non-Differentiating Features Only
- CSP Representation:
 - Constrained Objects
 - Integer, Boolean, Real, Set
 - Backtrackable Objects
 - Constraints
 - Basic Arithmetic and Logical
 - Global (the most popular only)
 - Constraint combinations and User-defined constraints
- CSP Resolution:
 - Search Strategies (aka Goals) – the most popular only
 - Solution Iterators
 - Goal combinations, User-defined goals, Backtrackable Actions

What Not To Standardize

- Implementation mechanisms for
 - Constraint propagation
 - Domain representations
 - Backtracking and reversibility
 - Major binary and global constraints
 - Goal execution (?)
- Define hooks for unique, solver specific features
- Standard should not limit innovation

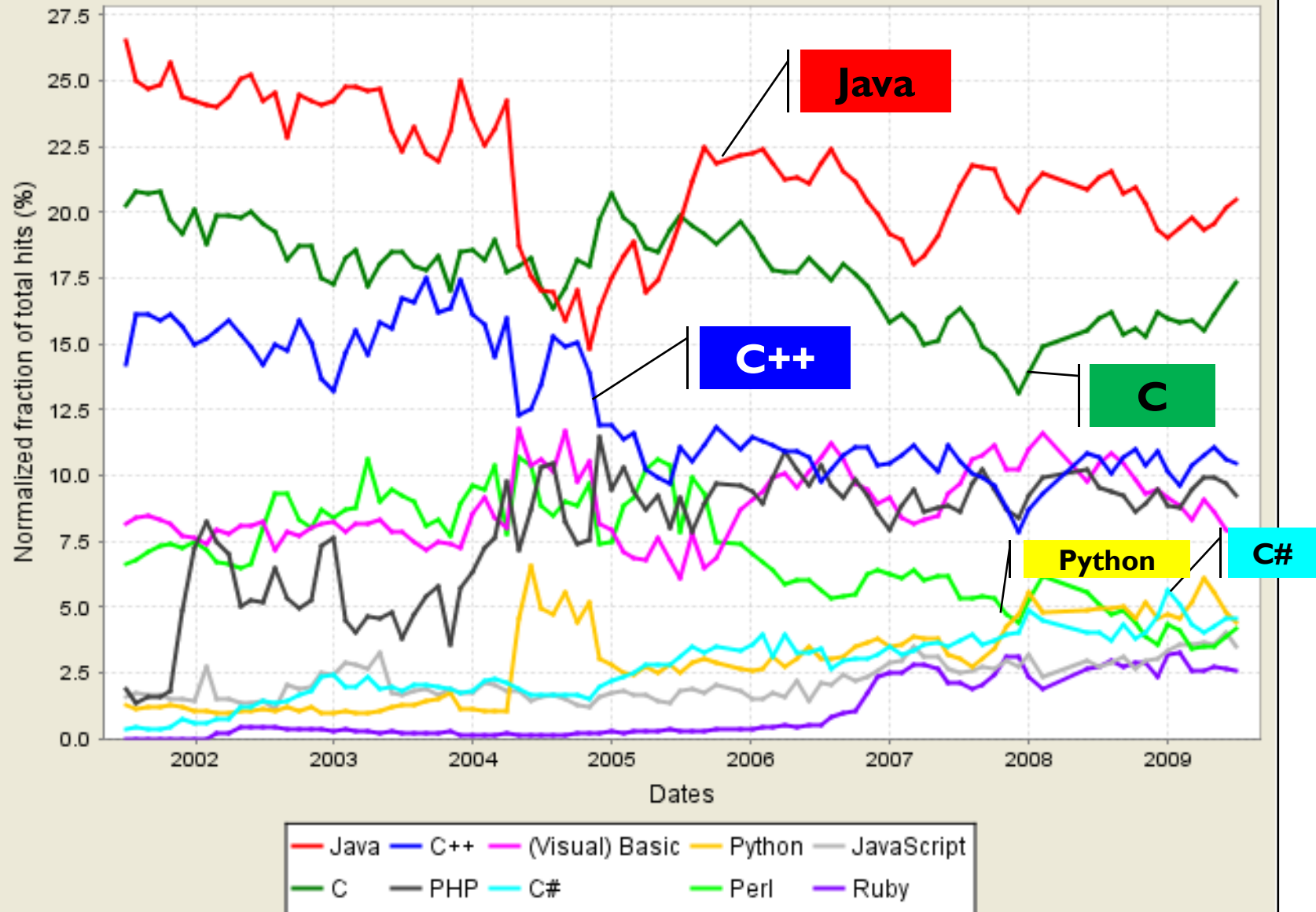
Target Languages

- Main-Stream programming languages
 - Java
 - C#
 - C++
- Unified XML Schema
 - XCSP, FlatZinc, SWRL/CIF, ..
- Unified CP Modeling Language (?)
 - Based on MiniZinc, OPL, Comet, ...
 - Python
 - Prolog

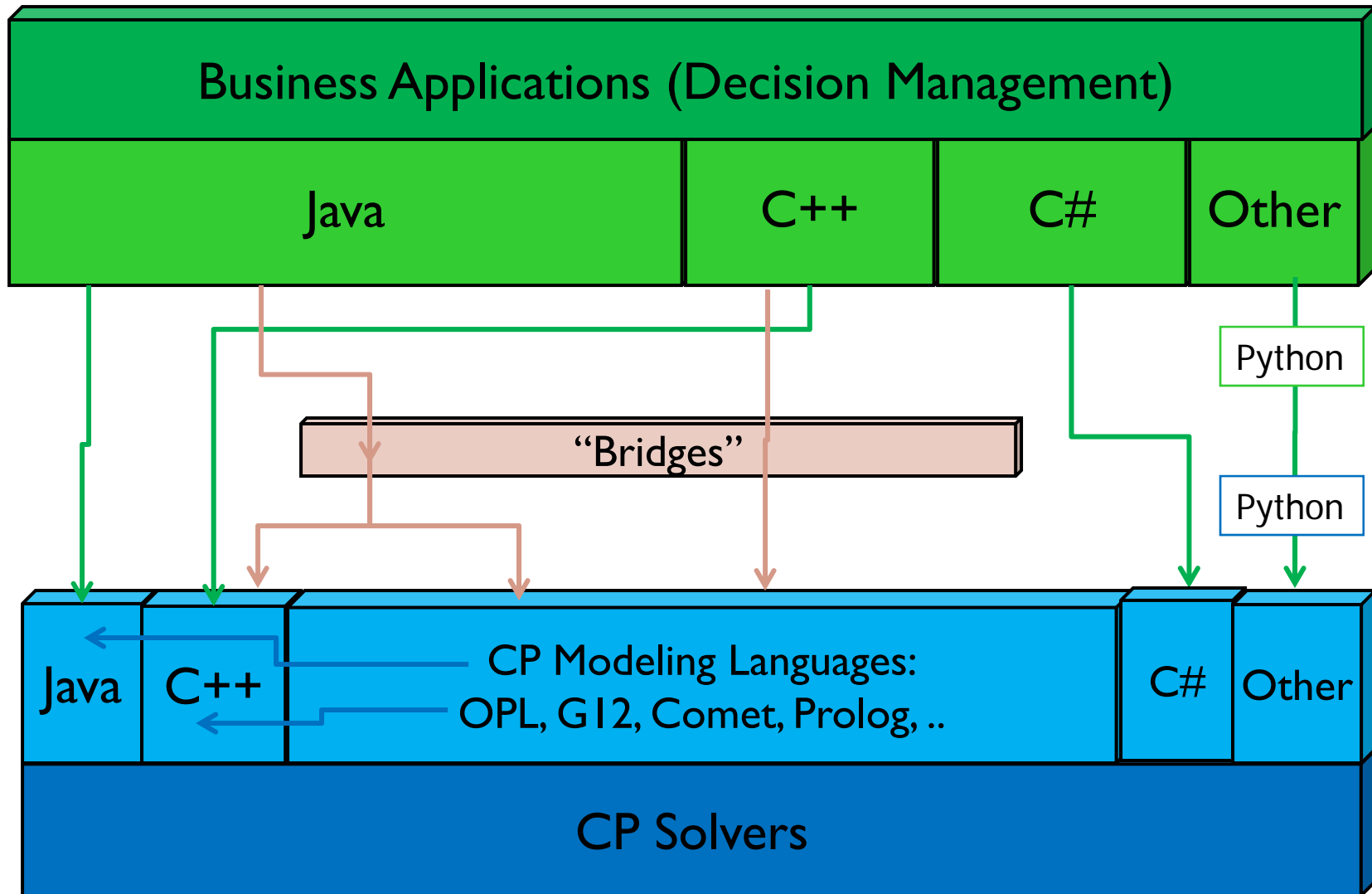
Programming Languages Index for July 2009

Tiobe Programming Community Index

Source: www.tiobe.com



Languages for Business Applications and for CP



JSR 331: Java CP API

- First but still one of many possible standard interfaces
- Similar to C++ and C# APIs
- Reference Implementations and Technology Compatibility Kit
- Join JSR331 Expert Group: www.jcp.org

Current CP Standardization Status

- Visit www.cpstandards.org
 - Straw-man proposals and open issues
 - Discussion Forum
 - CSP examples in different languages
 - Current Java CP API
- TimeLine
 - JSR 331
 - First Public Review: Nov. 2009
 - Final Release: April 15, 2010
 - Maintenance review: June 30, 2010
 - Other standards: ?
- Need constructive CP community input