Sravan Bhamidipati	Tester
Michael Deeringer	Language Guru
Fang-Hsiang Su	System Architect
Jiacheng Yang	System Integrator
Chun-Yu Tsai	Project Manager

gramola

a graph modeling language

2013 Spring PLT	Team 4

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Outline

Introduction: Why and How Gramola

Language Highlights

Project Management

Gramola Translator Architecture

Runtime Environment

Test Plan

Demo

Conclusions

Columbia University
in the City of New York

Why Gramola?

- Graph Theory
  - An important subject of modern science
  - Applied in numerous domains: social networks
- Many languages support Graph, but they
  - are not developer-friendly: Longer learning curve
  - focus on limited functionalities like graph DB or draw
- Developers need the power of

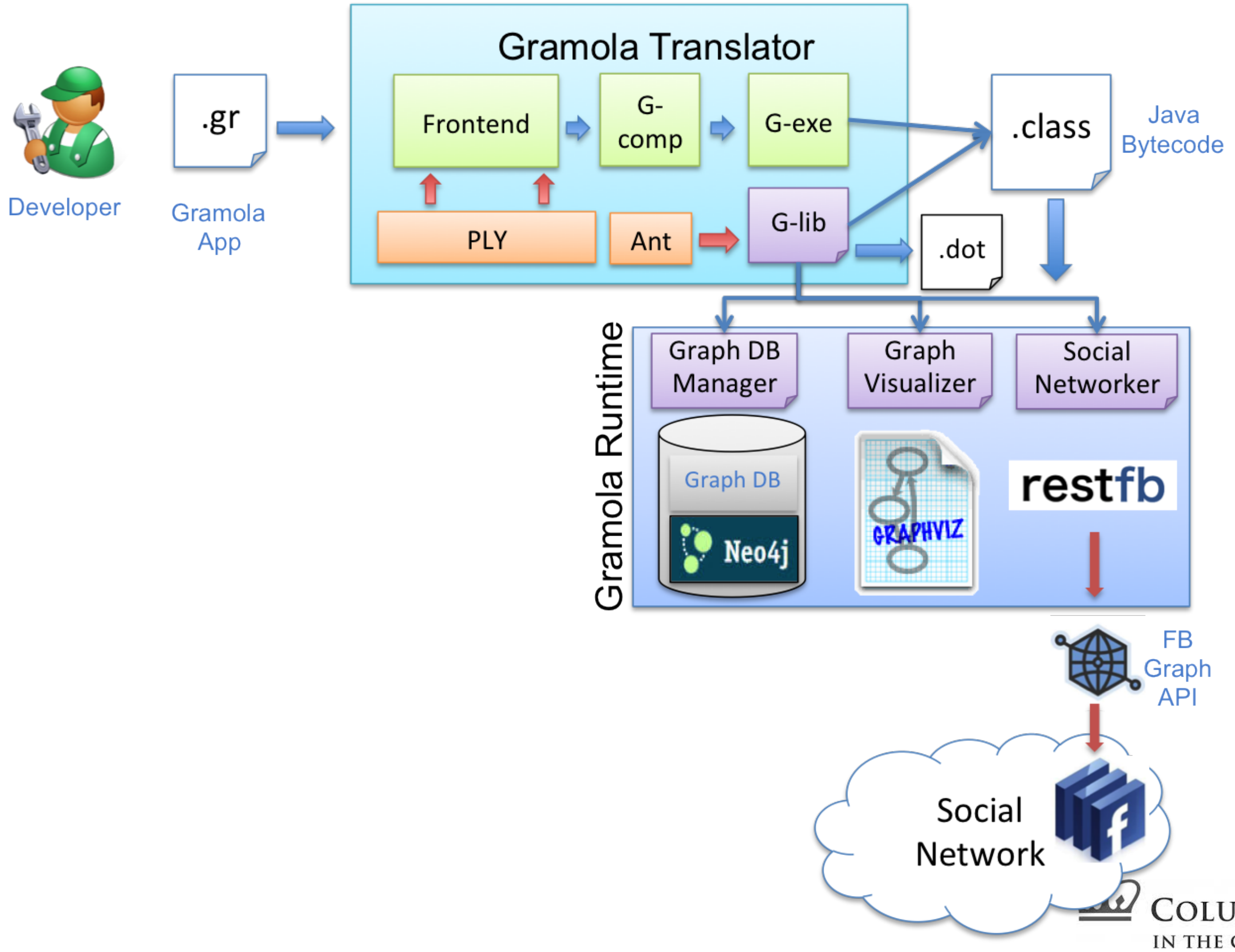
Graph-Native

Intuitive

Easy

Extensible

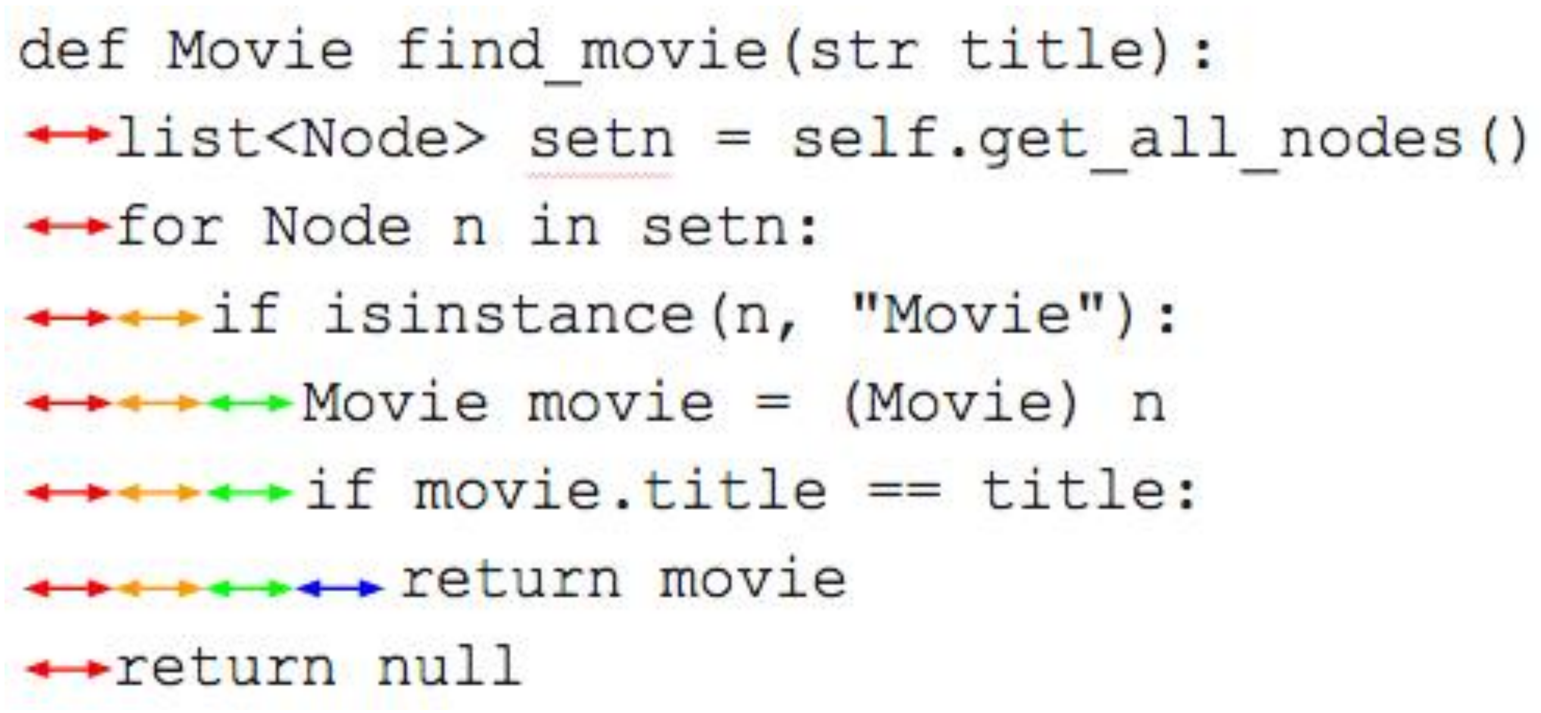
Gramola Module Overview



COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Language Highlights

Indentation-based blocks, logical lines



```
def Movie find_movie(str title):
	list<Node> setn = self.get_all_nodes()
	for Node n in setn:
		if isinstance(n, "Movie"):
			Movie movie = (Movie) n
			if movie.title == title:
				return movie
	return null
```

Language Highlights

Many "general-purpose" features

```
def void main():
  for object j in [2, 3]:
      int i = (int) j
      if i <= 3 and i > 2:
           print "i is less than 3"
      elif i + 1 <= 3:
           print "i + 1 less than 3"
```

Language Highlights

Classes, inheritance, namespaces

```
class Actor(Node):
  str name
  def Actor __init__(dict<str,str> dd,
                         str actorname):
      Node(dd)
      self.name = actorname
```

Columbia University
in the city of New York

Language Highlights

Built-ins

```
str token1 = "login_token"
Graph fb1 = get_fb(token1)
draw(fb1, "name", "type")
dump(fb1, "PLT")
```

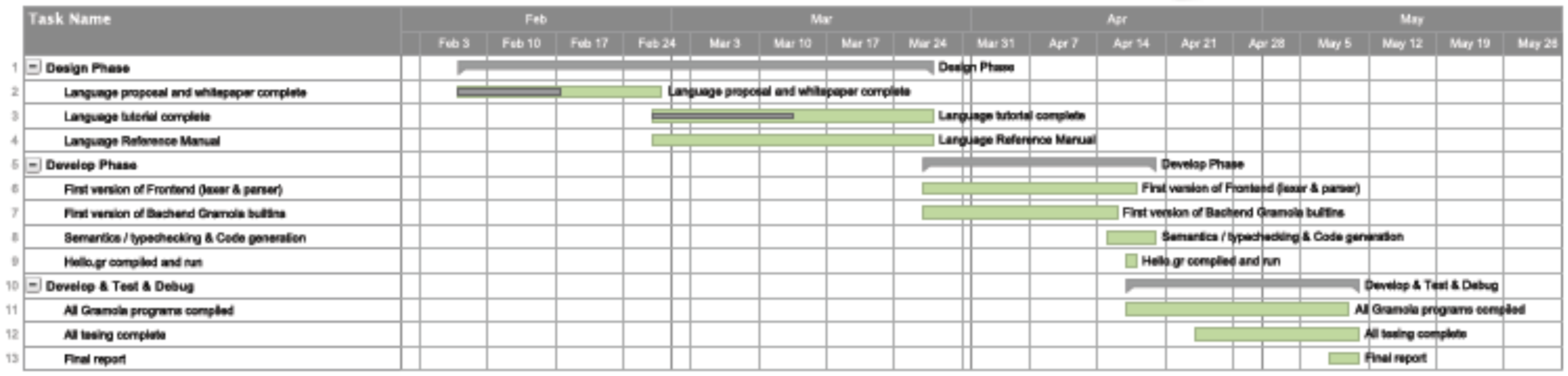
Project Management

- Weekly Scheduled Meetings
- Google Drive
  - Document management
- Googlegroups
  - Announcement
  - Meeting Agenda
  - Coordinating remote work
- GitHub
  - Gramola version control

Columbia University
in the City of New York

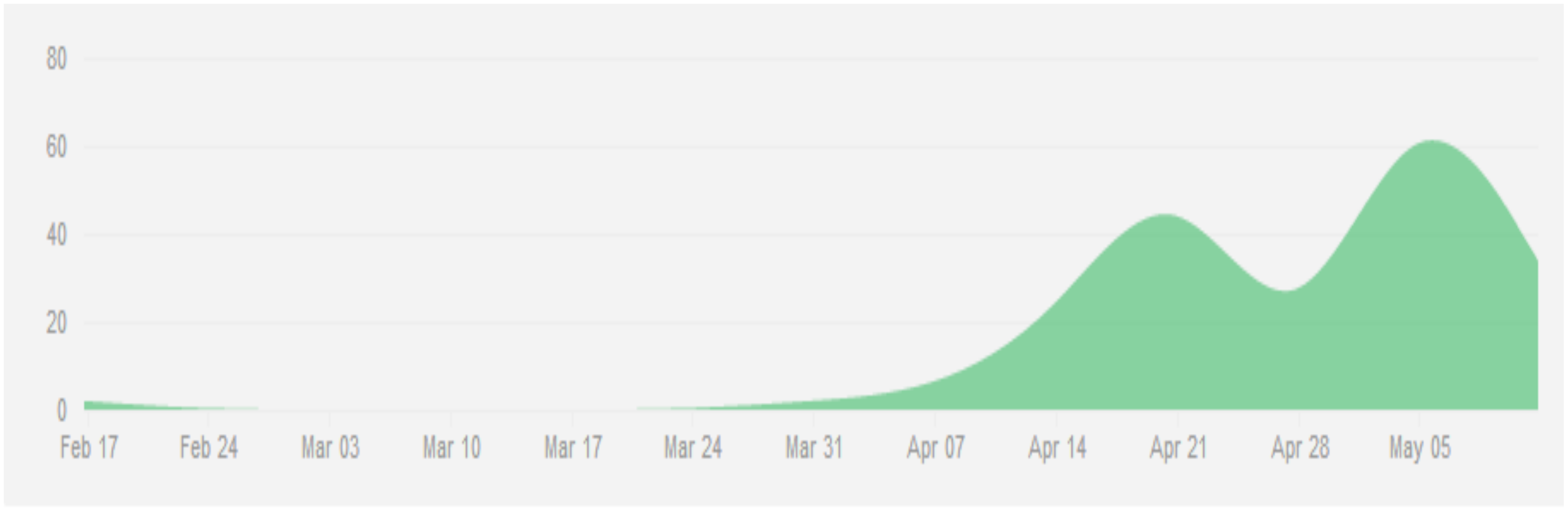
Project Management

- Iterative and incremental project planning
- Project timeline
    - Gantt Chart



Project Management

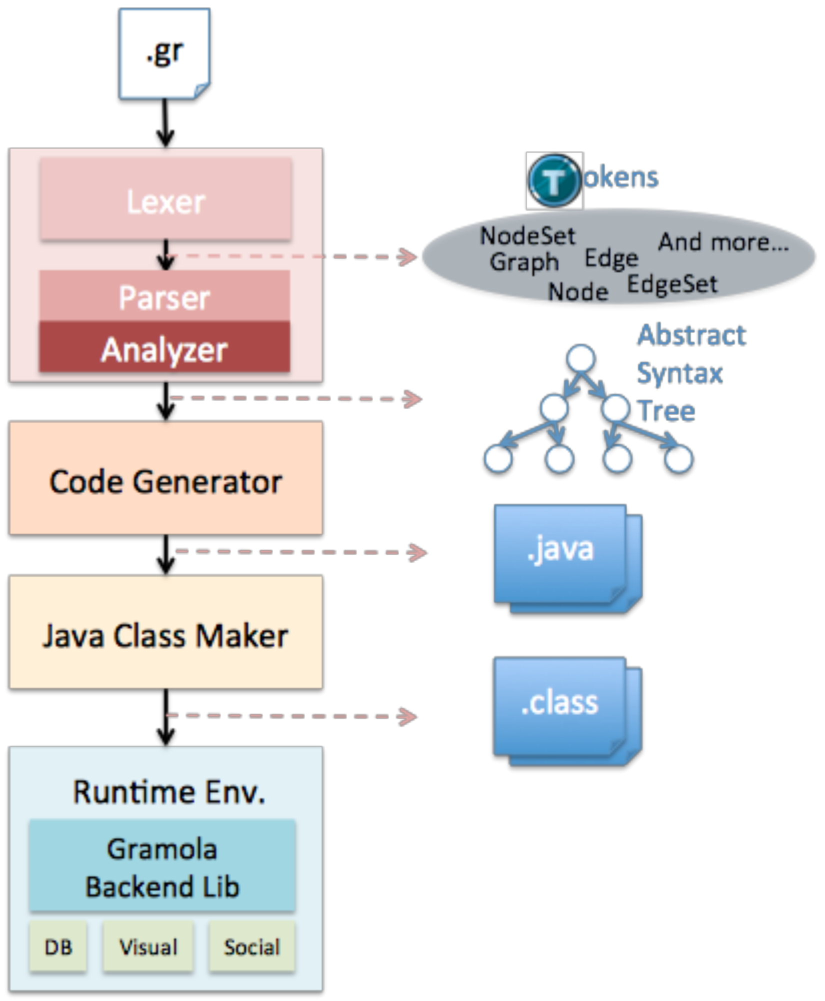
- GitHub commits by days



Development Environment

- Frontend: PLY/Python
- Backend: Java
- Version Control: Git with Github hosting
- **gcompile**

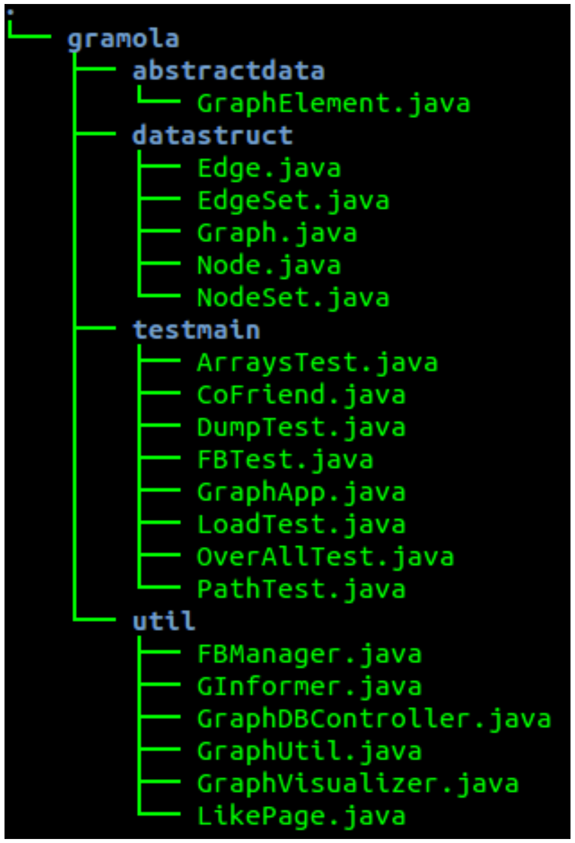
Gramola Translator



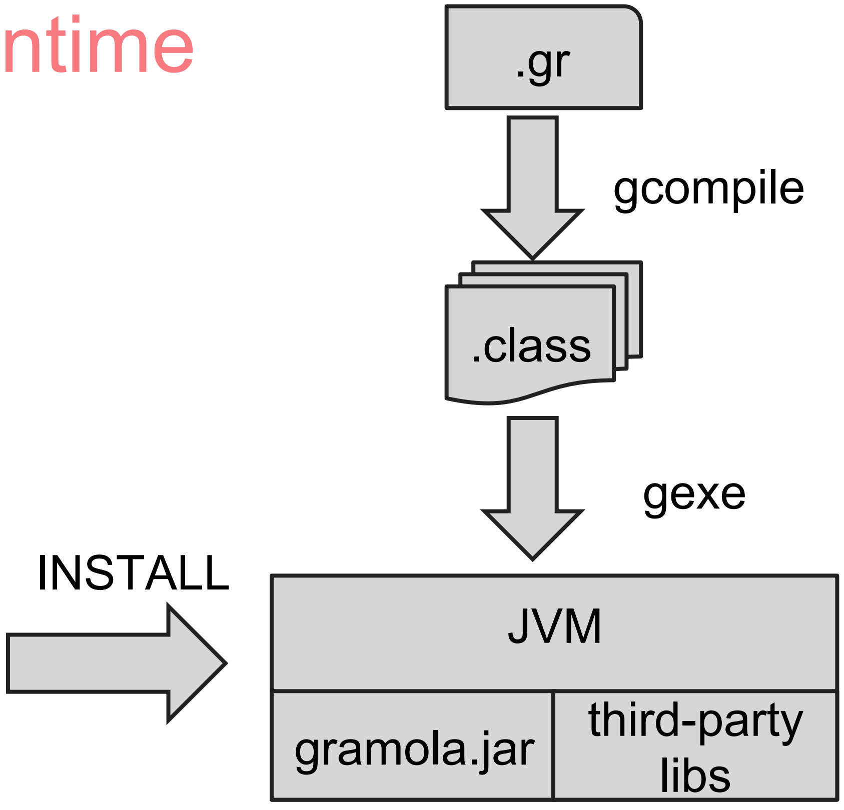
COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Runtime Environment

- Java-powered runtime



```
.
└── gramola
    ├── abstractdata
    │   └── GraphElement.java
    ├── datastruct
    │   ├── Edge.java
    │   ├── EdgeSet.java
    │   ├── Graph.java
    │   ├── Node.java
    │   └── NodeSet.java
    ├── testmain
    │   ├── ArraysTest.java
    │   ├── CoFriend.java
    │   ├── DumpTest.java
    │   ├── FBTest.java
    │   ├── GraphApp.java
    │   ├── LoadTest.java
    │   ├── OverAllTest.java
    │   └── PathTest.java
    └── util
        ├── FBManager.java
        ├── GInformer.java
        ├── GraphDBController.java
        ├── GraphUtil.java
        ├── GraphVisualizer.java
        └── LikePage.java
```



Runtime Environment

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Runtime Environment

- Scripts
  - INSTALL: one-click configuration of the runtime environment. For internal use and first-time user installation
  - gcompile: compile .gr ->.java -> .class
    - ./gcompile hello.gr Hello
  - gexe: invoke JVM to link compiled user program (.class), the gramola library (gramola.jar) and other supporting libraries
    - ./gexe Hello

Runtime Environment

- The Gramola library
  - Built-in data structures for graphs, e.g. Graph, Node, Edge, etc
  - Implementations of syntax sugars, e.g. initialize dictionaries with arbitrary number of key-value pairs
  - Converters/Drivers to connect to third-party libraries for advanced features, e.g. graph persistence, graph visualization

Test Plan

Pylint

check your Python code before running it

FindBugs

because it's easy

- Fuzz testing: Lexer
- End-to-end automated testing: To localize bugs to a specific compiler phase.
- About 900 LOC of syntactically and semantically valid Gramola programs to test every keyword, operator, built-in, data type, data structure, programming construct through every phase of the compiler.
- Less focus on error-handling.

Columbia University in the City of New York

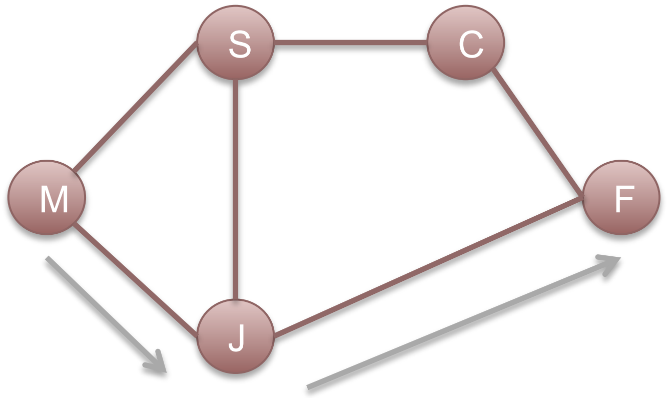
Test Results

Phase	Passed	Failed
Lexical Analysis	35	0
Syntax & Semantic Analysis	31	4
Code Generation	31	4
Compilation	30	5
Execution	30	5

Columbia University
IN THE CITY OF NEW YORK

Demo

1. IMDB
   a. Inheritance
   b. Loop
   c. Control Flow
2. Common Friends on Facebook
   a. User-defined class
   b. get_fb: Real-time data retrieval from Facebook
   c. dump: Graph data storage in Graph DB
   d. draw: Graph object=>dot => Graph visualization
3. More time?
   a. get_shortest_path: Shortest path finding
   b. Actually we have about 10+ Gramola apps!!

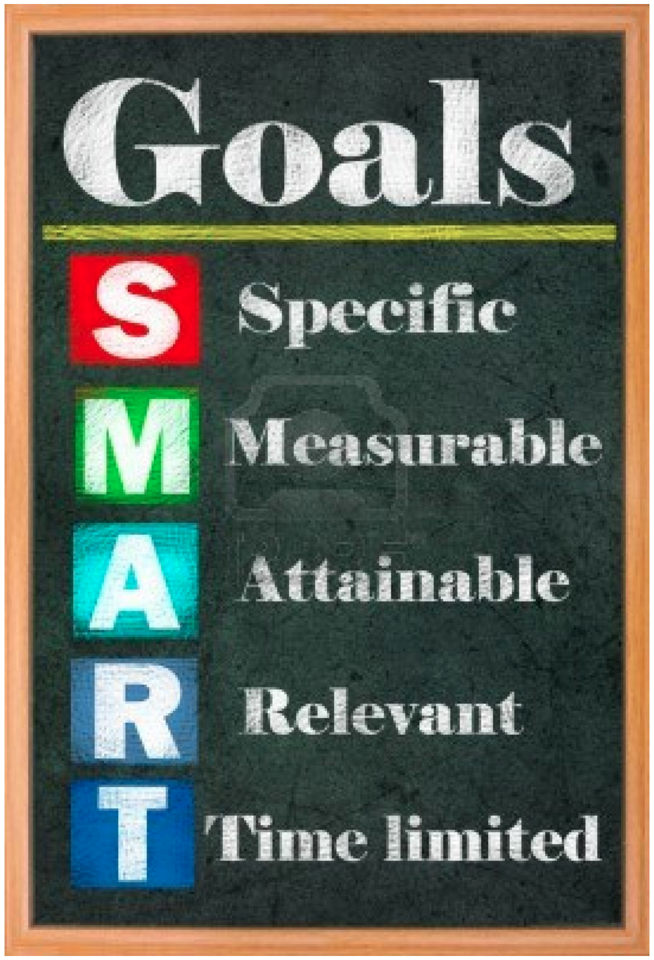


COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Conclusions

- We're proud of...
  - Gramola graph-native features (e.g., connection to FB), extensibility (class inheritance)
- What worked well...
  - gcompile
  - git version control
- Lessons we learnt
  - Start testing immediately after feature implemented!
  - We should plan for suitable scoping at the beginning!

We Can Do Better!



Measurable Progress

Relevant Workload

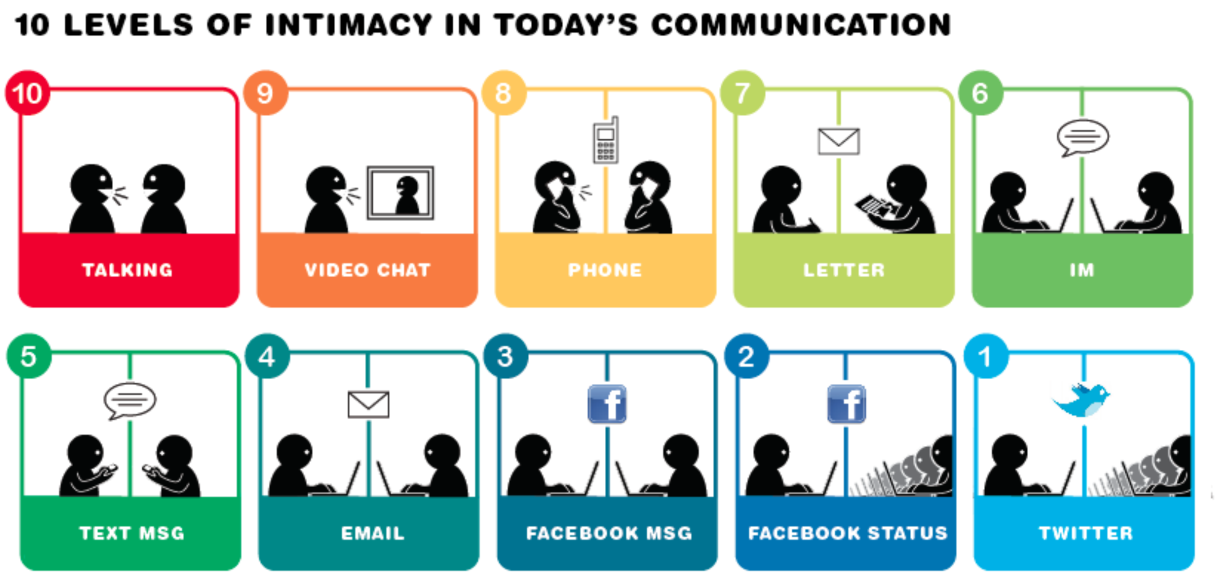
Specific Features

Attainable Target

Yes, time is never enough

Columbia University
IN THE CITY OF NEW YORK

Work Together



```
while work_in_same_place(team4):
    productivity[team4] += 1
```

COLUMBIA UNIVERSITY
IN THE CITY OF NEW YORK

Thanks!

Title logo taken from http://www.etringita.com/pequeneces/2010/05/05/gramola/
RCA logo redux taken from http://kayleighmahon.wordpress.com/2012/09/

Columbia University
In the City of New York