The ERATO *Systems Biology Workbench*: Enabling Interaction and Exchange Between Tools for Computational Biology

Michael Hucka, Andrew Finney, Herbert Sauro, Hamid Bolouri

ERATO Kitano Systems Biology Project
California Institute of Technology, Pasadena, CA, USA

Principal Investigators: John Doyle, Hiroaki Kitano

Collaborators:

Adam Arkin (BioSpice), Dennis Bray (StochSim),
Igor Goryanin (DBsolve), Andreas Kremling (ProMoT/DIVA),
Les Loew (Virtual Cell), Eric Mjolsness (Cellerator),
Pedro Mendes (Gepasi/Copasi), Masaru Tomita (E-CELL)

Motivations

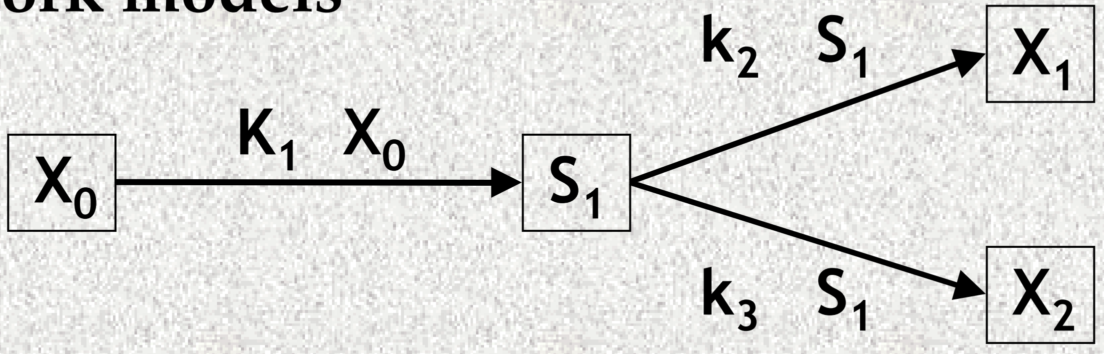
- **Observation:** ***proliferation of software tools***
- **No single package answers all needs**
  - Different packages have different niche strengths
  - Strengths are often complementary
- **No single tool is likely to do so in the near future**
  - Range of capabilities needed is large
  - New techniques (new tools) evolve all the time
- **Researchers are likely to continue using multiple packages for the foreseeable future**
- **Problems with using multiple packages:**
  - Simulations & results often cannot be shared or re-used
  - Duplication of software development effort

Project Goals & Approach

- **Develop software & standards that**
  - Enable sharing of simulation & analysis software
  - Enable sharing of models
- **Goal: make it easier to share tools than to reimplement**
- **Two-pronged approach**
  - Develop a common model exchange language
    - SBML: Systems Biology Markup Language
  - Develop an environment that enables tools to interact
    - SBW: Systems Biology Workbench

Systems Biology Markup Language (SBML)

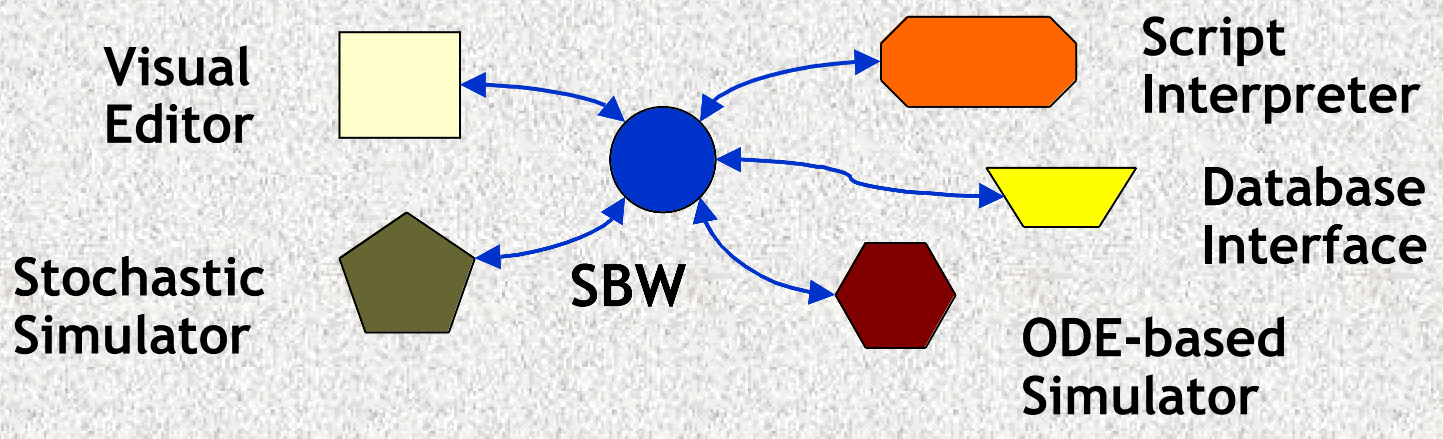
- **Biochemical network models**



- **A model is described using a list of components:**
  - Beginning of model definition
    - » List of unit definitions (optional)
    - » List of **compartments**
    - » List of **species**
    - » List of parameters (optional)
    - » List of rules (optional)
    - » List of **reactions**
  - End of model definition

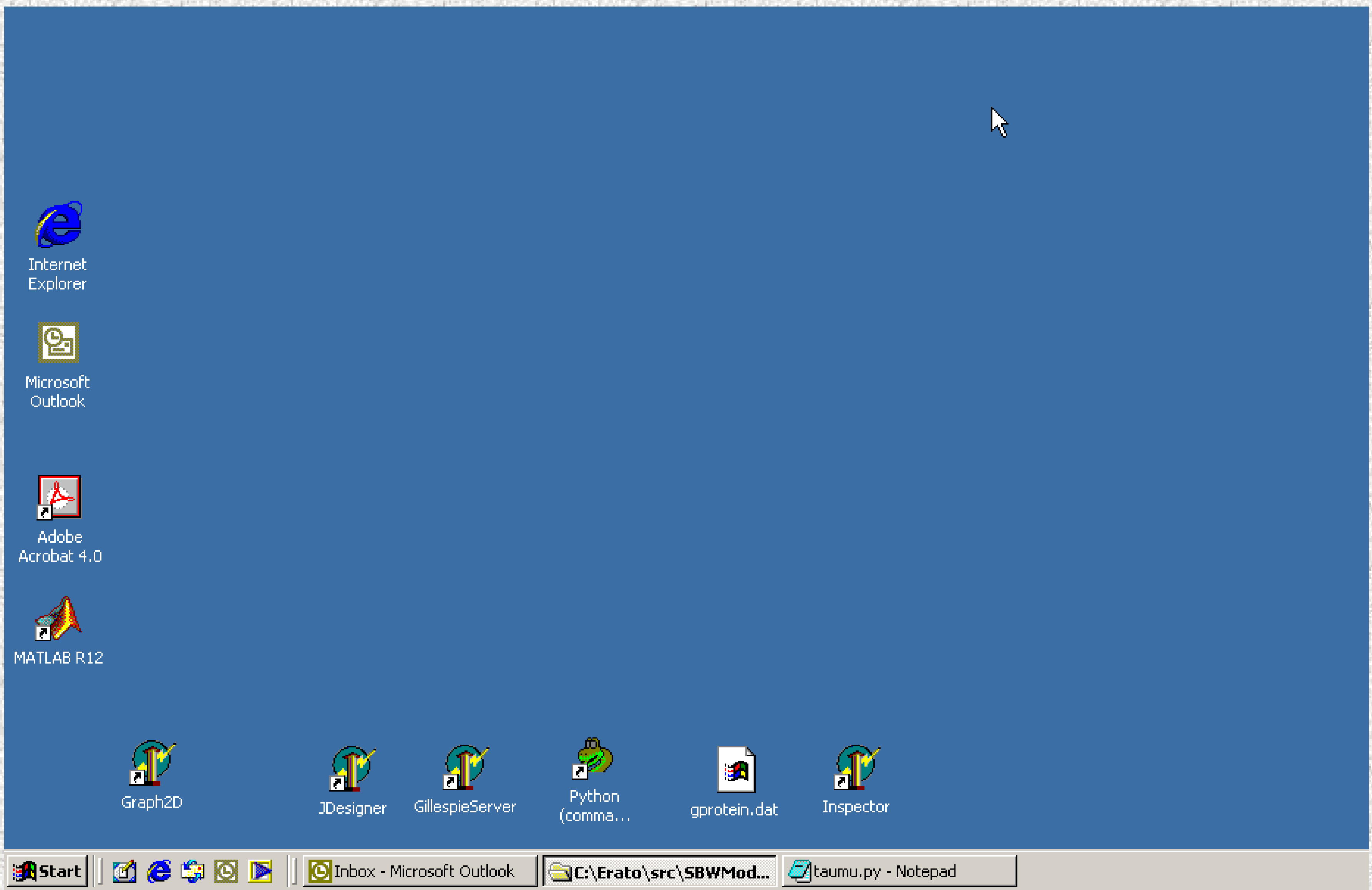
Systems Biology Workbench (SBW)

- **Simple framework for enabling application interaction**
  - Free, open-source (LGPL)
  - Portable to popular platforms and languages
  - Small, simple, understandable



- **From the user's perspective, SBW is invisible**

SBW From the User's Perspective

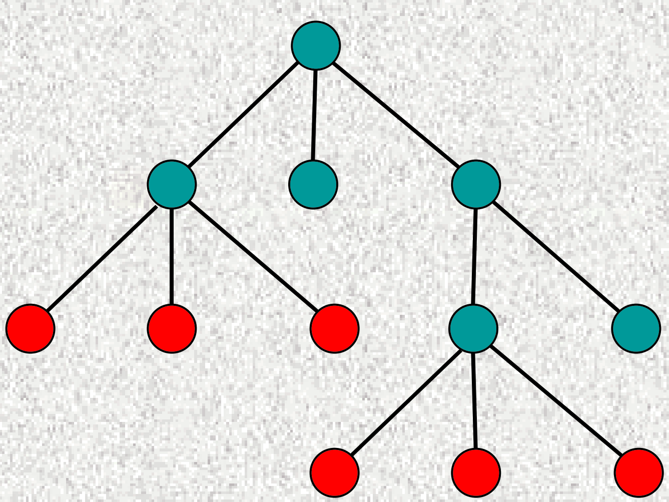


From the Programmer’s Perspective

- **Simple, lightweight, message-passing architecture**
  - Cross-platform compatible & language-neutral
- ***Modules*** **are separately-compiled executables**
  - A module defines *services* which have *methods*
  - SBW native-language libraries provide APIs
    - C, C++, Java, Delphi, Python available now
    - … but can be implemented for any language
- ***SBW Broker*** **acts as coordinator**
  - Remembers services & modules that implement them
  - Starts modules on demand
    - Broker itself is started automatically
  - Notifies modules of events (startup, shutdown, etc.)

The SBW Broker's Registry

- **Registry records information about modules**
  - Module name
  - How to start module
  - What services the module provides
  - The categorization of those services
- **Hierarchy of service categories**



Service Categories (Interface Hierarchy)

Services (Interfaces)

Example of Service Categories

Service	Methods

Simulation	`void loadModel(string SBML)`
	`void setStartTime(double time)`
	`void setEndTime(double time)`
	`void run()`

	+

ODESimulation	`void setIntegrator(int method)`
	`void setNumPoints(int num)`

Service Categories Group Applications

- **Clients can be written to interact with classes of modules in a generic way**

(Java)

```
interface ODESimulation {
     void loadModel(string SBML)
     void setStartTime(double time)
     void setEndTime(double time)
     void run()
     void setIntegrator(int method)
     void setNumPoints(int num)
}
```

- **User menus can be grouped by categories**
- **Need help from community to define common categories of interfaces**

Why?

- **Why not use CORBA?**
  - Complexity, size, compatibility
  - Could not find fully-compliant open-source CORBA ORB that supports all required programming languages
  - SBW scheme does not require a separately compiled IDL
  - But: planning to have gateway between CORBA & SBW
- **Why not use SOAP or XML-RPC?**
  - Performance, data type issues, implementation quality
- **Why not Java RMI?**
  - Java-specific
- **Why not COM?**
  - Microsoft-specific, low portability

SBW Status & Future

- **Beta release: http://www.cds.caltech.edu/erato**
  - **Java, C, C++, Delphi, Python libraries**
  - **Windows & Linux**
  - **Developer's manuals & tutorials, examples**
  - **Modules:**
    - SBML Network Object Model
    - MATLAB model generator
    - Plotting module
    - Jarnac ODE simulator
    - Optimization module
    - Stochastic simulator
    - JDesigner visual editor
- **Spring 2002: production release 1.0**
  - Perl and C# libraries
  - Secure distributed operation
  - CORBA gateway
  - More modules: Bifurcation analysis, Gillespie "Tau-Leap"