MIT iLabs:
Carnegie Initiation Meeting
Makerere University

June 26, 2005

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

mit iCampus

iLab Design Goals

- Scaling usage of online labs to a large number of users
- Encouraging researchers and universities to share their labs online
- Single sign on to labs at multiple universities
- Freeing lab owner/operator from administration (i.e. authentication, authorization, storage of results, archiving of data, etc.) of users from other universities
- Allowing universities with diverse network infrastructures to interoperate and share resources

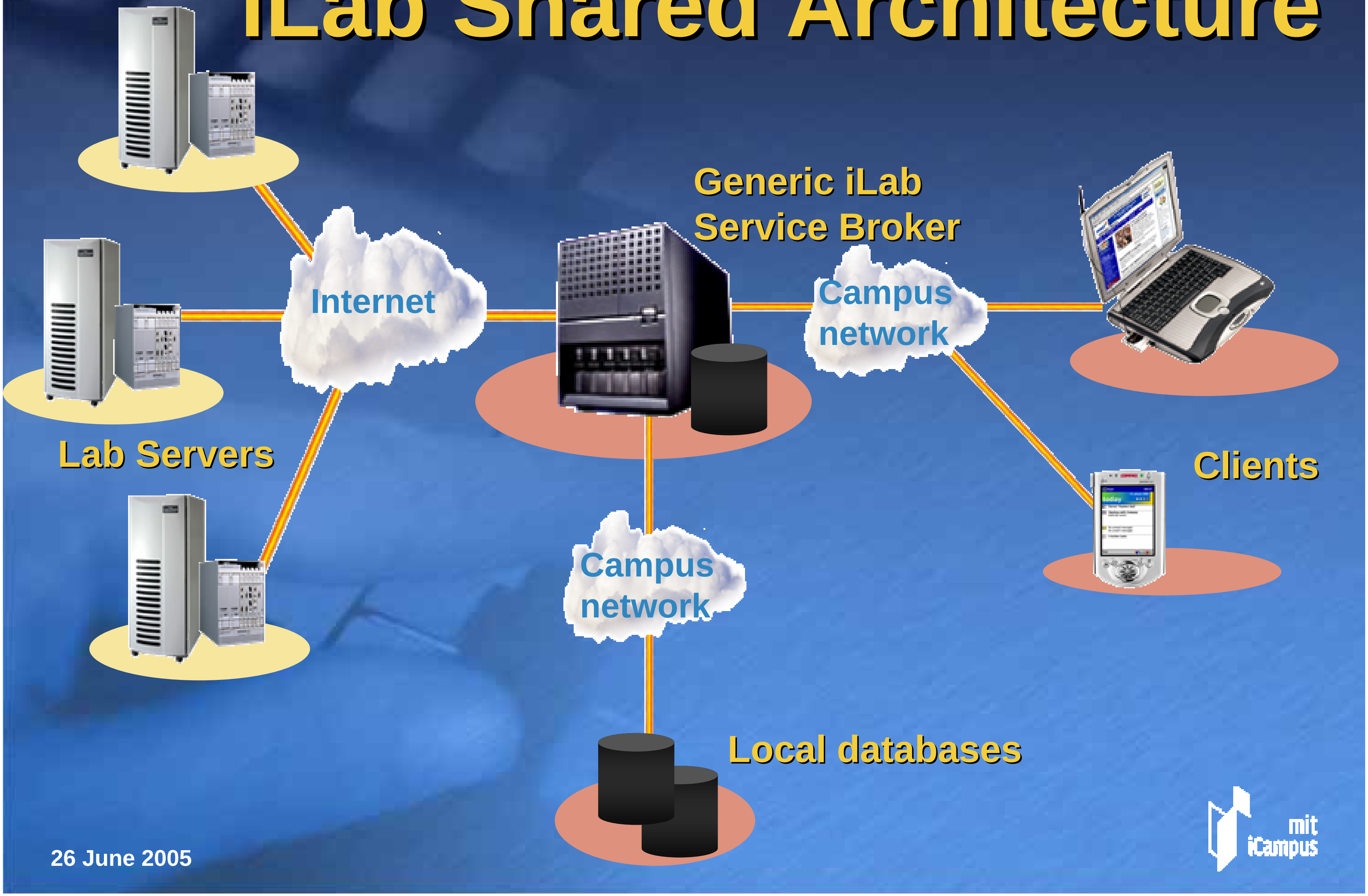
Project Boundaries

- Our architecture doesn’t deal with specific hardware and software interfaces to lab equipment
- Our architecture is intended to be compatible and complementary with commercial software such as National Instruments LabView and analysis packages like Matlab

iLab Generic Services

- User authentication (and registration)
- User authorization and credential (group) management
- Experiment specification and result storage
- Lab access scheduling

iLab Shared Architecture



26 June 2005

The Case for Web Services

- Web services represent the latest version of an old concept -- they allow one computer to invoke a procedure (method) on another.
- They are platform and vendor independent (we have already successfully bridged a Java client ⇔ a Windows XP/.NET Service Broker ⇔ a Windows 2000 lab server (with NI GPIB).
- Web services are self-describing and offer the promise of runtime discovery.
- Because they are usually based on http that we all use to access the web, they work well with campus networks.
- The iLab Shared Architecture builds on top of the current generation of web services.

iLab Experiment Typology, 1

3 Waves of Development

- Batched Experiments (2003-2005):
  - The entire specification of the experiment is determined before execution begins.
  - The user need not remain online while experiment executes.
- Interactive Experiments (2004-2006):
  - The student client portrays virtual lab equipment (GUI).
  - The student can interact with experiment throughout its course.

iLab Experiment Typology, 2

3 Waves of Development

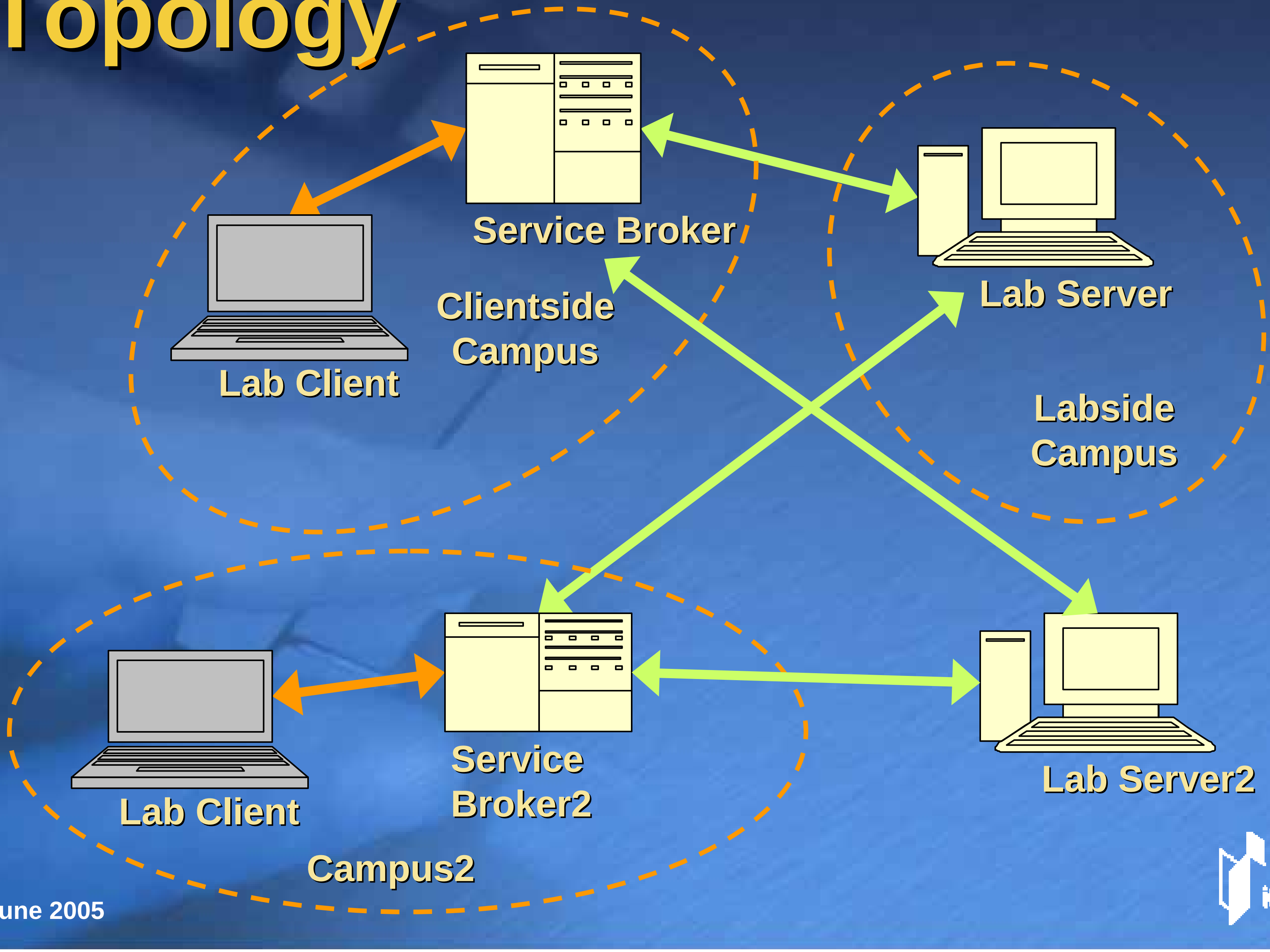
- Sensor Experiments (2005-2007?):
  - Publish and subscribe based architecture
  - Triggers and event-driven data monitoring
  - Flexible data analysis
  - Data archive

iLabs Design Strategy

Separate responsibilities of the lab provider from those of the teaching faculty

- The lab provider designs and makes the laboratory experiment available online in as effective a presentation as possible
- The teaching faculty register their own students, manage their accounts and result storage, and set course policy (e.g. can students collaborate)

Batched Experiment Topology



26 June 2005

mit iCampus

Service Broker Responsibilities

The Service Broker is a domain independent server that

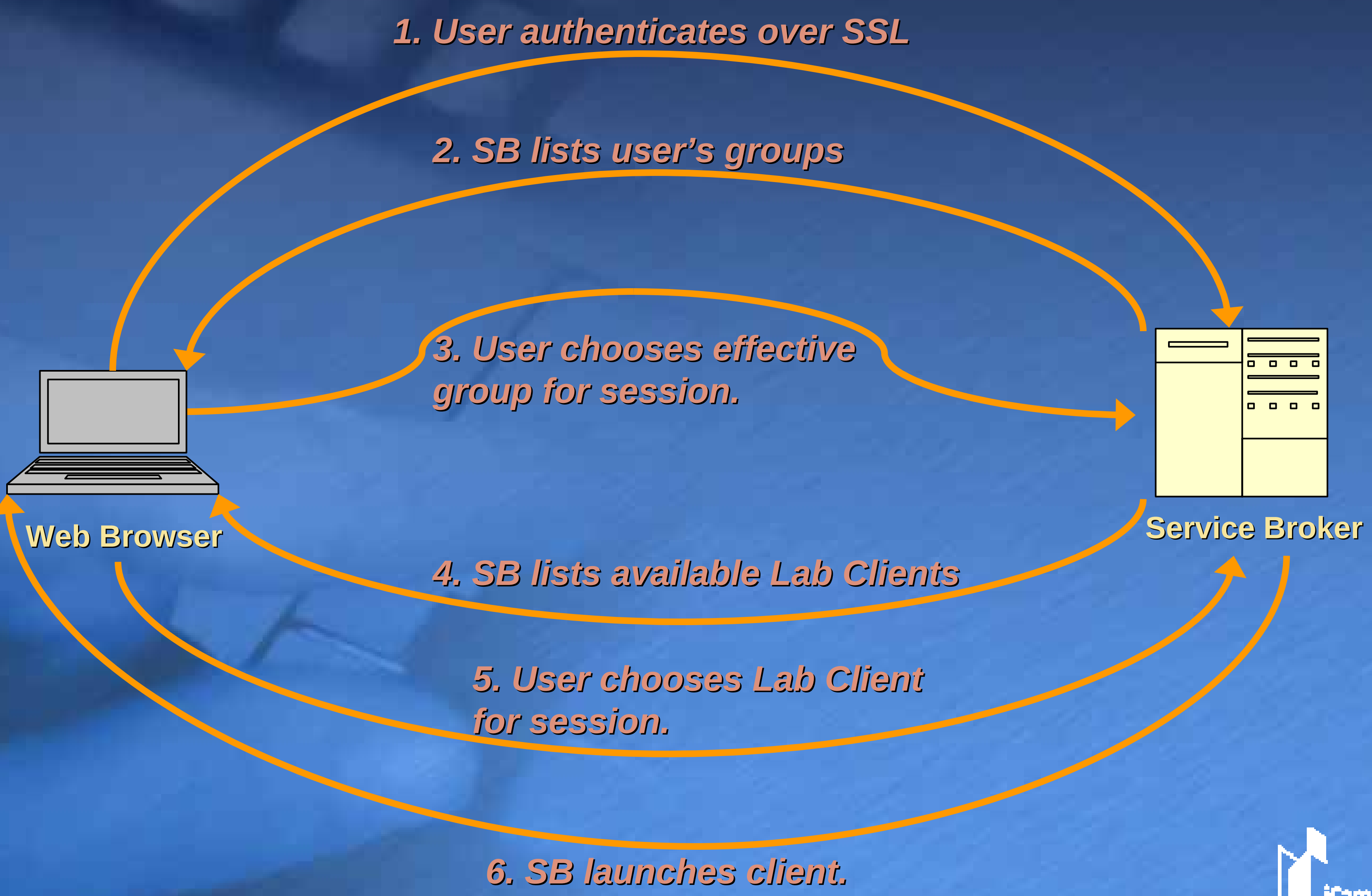
- stores and manages student experiment records;
- provides mechanism for but does not specify local campus course and privacy policy;
- authenticates users and transmits credentials *but not user IDs to Lab Server;*
- manages workflow between client and lab server

Lab Provider Responsibilities

The Lab Server team

- builds the lab server which must implement the methods of the *Service Broker to Lab Server Web Service API*;
- usually supplies the student lab client software, which must implement the methods of the *Client to Service Broker Web Service API;*

Student Web Session



1. User authenticates over SSL

2. SB lists user's groups

3. User chooses effective group for session.

4. SB lists available Lab Clients

5. User chooses Lab Client for session.

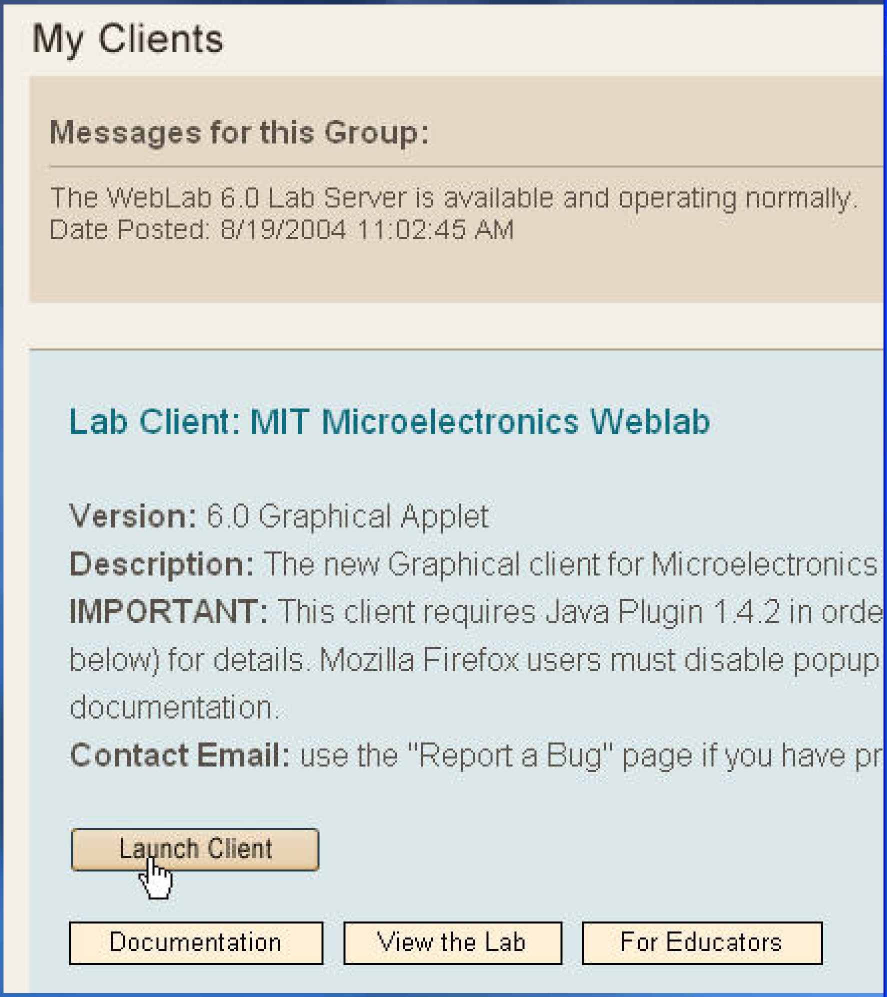
6. SB launches client.

mit iCampus

Student Service Broker Session Life Cycle

- The student contacts the Service Broker (SB) via a standard web browser.
- The student either
  - registers for a new account, or
  - authenticates himself to the Service Broker (current implementation offers ID/password over SSL)
- The SB lists the student’s group (class) memberships, and asks the student to choose an effective group for this session.
- The SB lists the lab servers/clients available to that effective group, and asks the student to choose a client
- The SB launches the lab client (often an applet) for the student.

Service Broker:
Launching the Client



My Clients

Messages for this Group:

The WebLab 6.0 Lab Server is available and operating normally.
Date Posted: 8/19/2004 11:02:45 AM

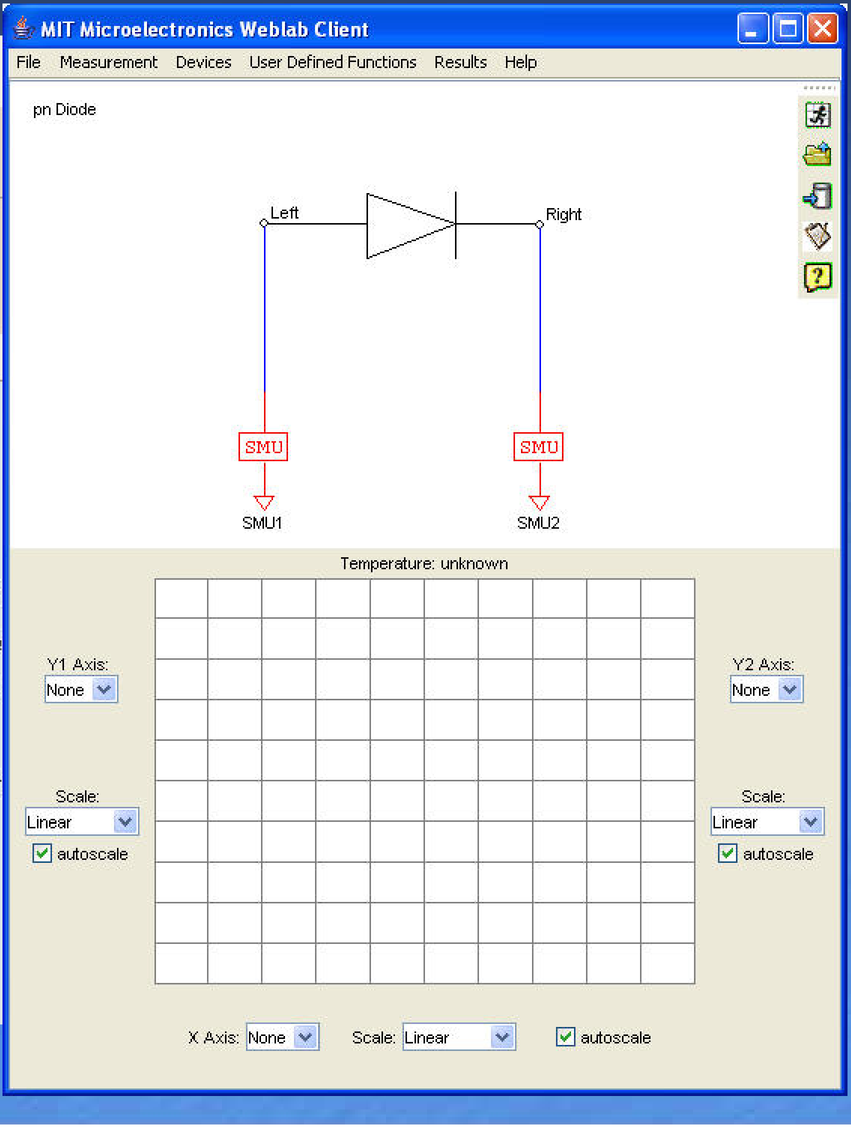
Lab Client: MIT Microelectronics Weblab

Version: 6.0 Graphical Applet

Description: The new Graphical client for Microelectronics

IMPORTANT: This client requires Java Plugin 1.4.2 in orde below) for details. Mozilla Firefox users must disable popup documentation.

Contact Email: use the "Report a Bug" page if you have pr



Y1 Axis: None

Scale: Linear

☑ autoscale

Y2 Axis: None

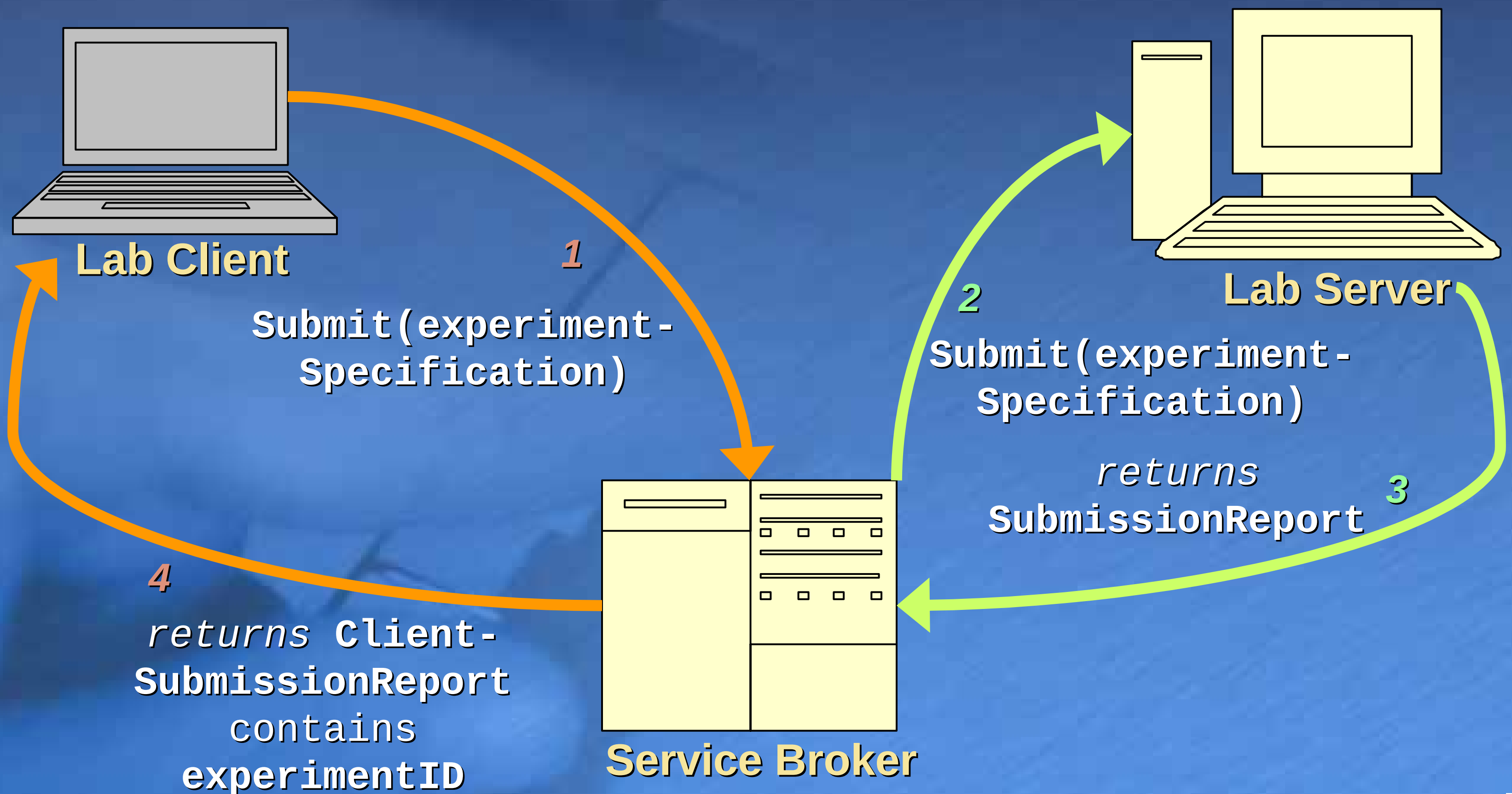
Scale: Linear

☑ autoscale

X Axis: None Scale: Linear ☑ autoscale

26 June 2005

Batched Experiment Submission
Web Service Calls



Lab Client

Lab Server

Service Broker

1

`Submit(experiment-Specification)`

2

`Submit(experiment-Specification)`

returns `SubmissionReport`

3

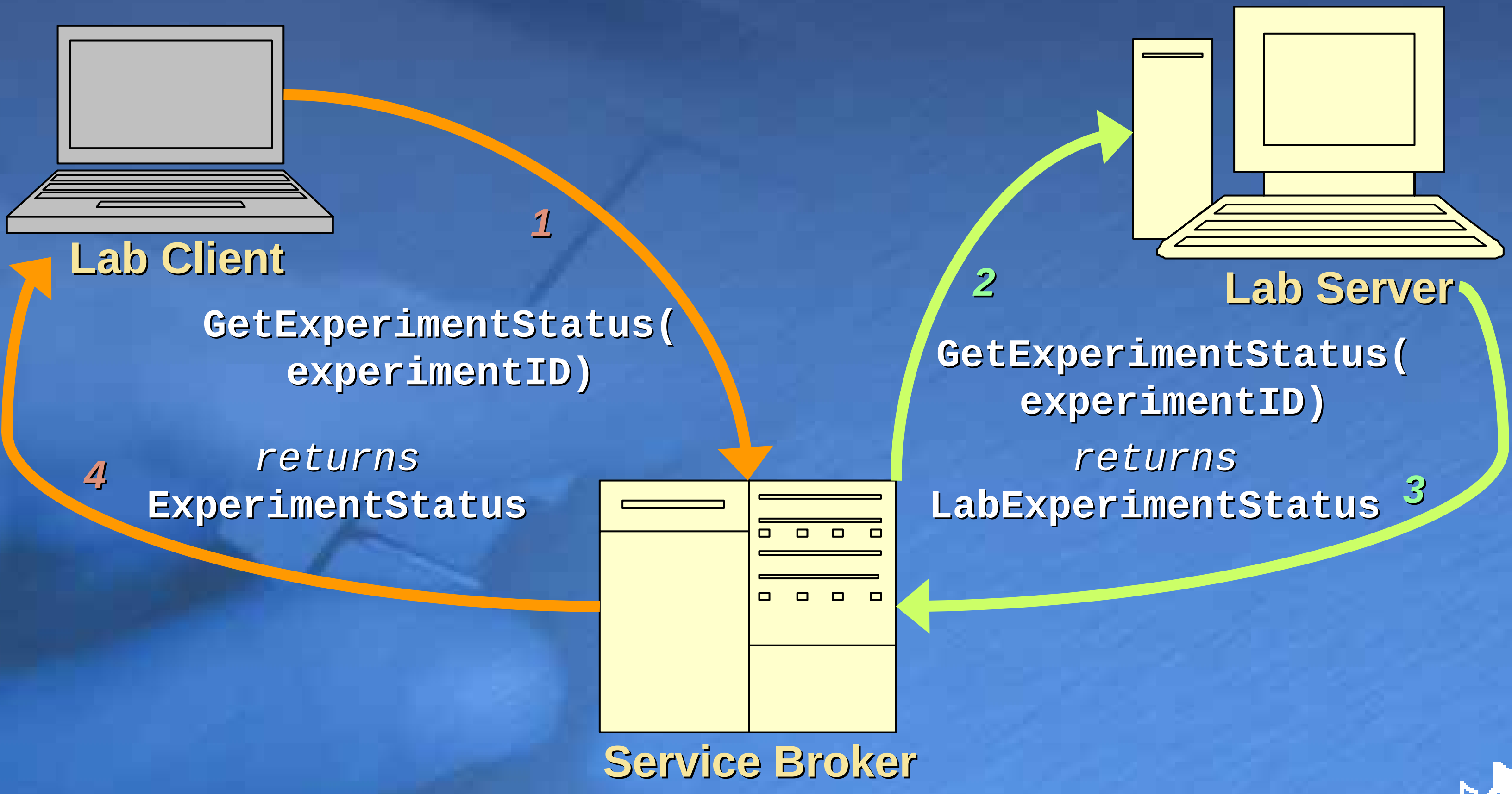
4

returns `Client-SubmissionReport` contains `experimentID`

26 June 2005

Batched Experiment Status Checking

Web Service Calls



Lab Client

1

GetExperimentStatus(
experimentID)

returns
ExperimentStatus

4

2

Lab Server

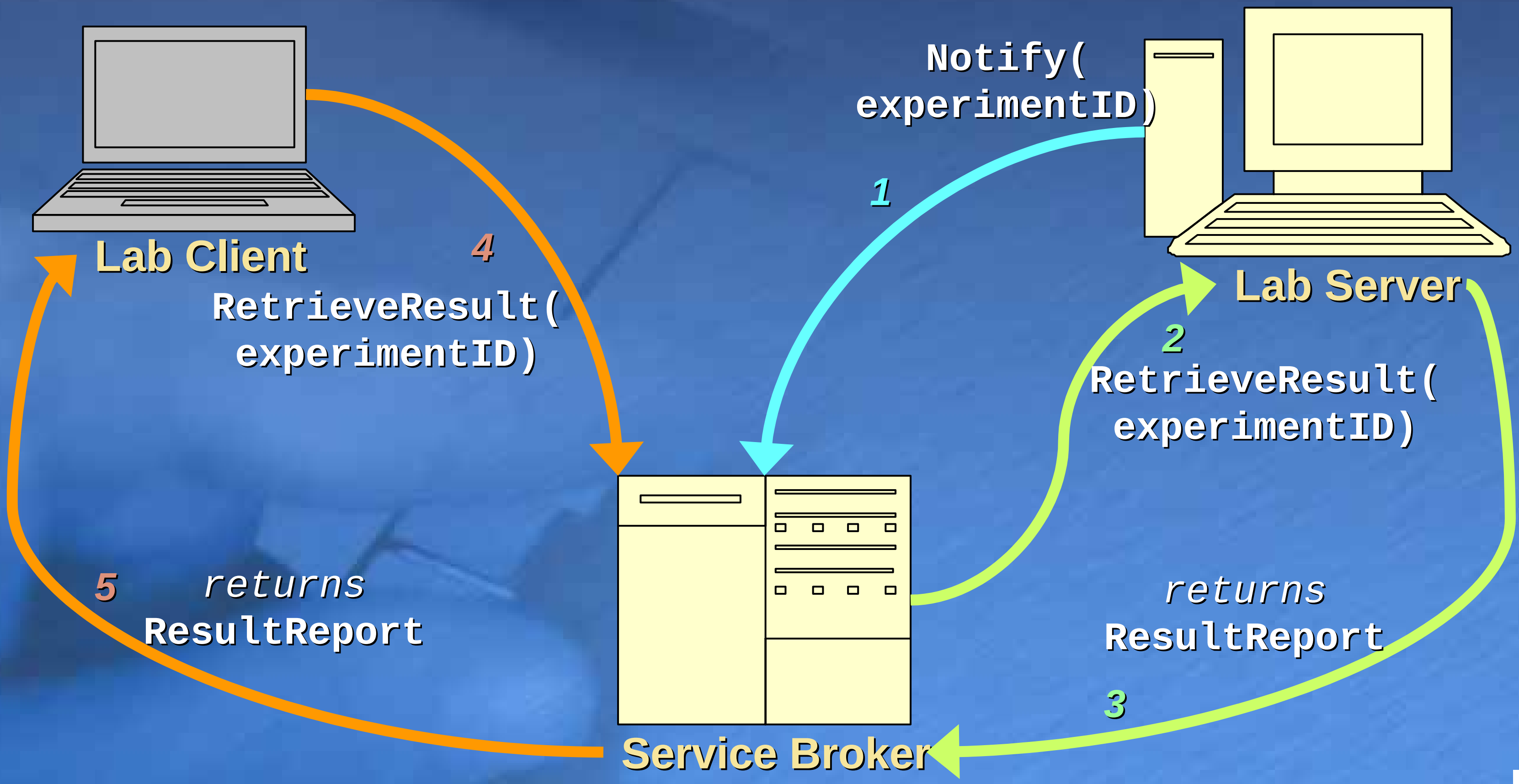
GetExperimentStatus(
experimentID)

returns
LabExperimentStatus

3

Service Broker

Batched Experiment Result Retrieval
Web Service Calls



Lab Client

Lab Server

Service Broker

1 Notify(experimentID)

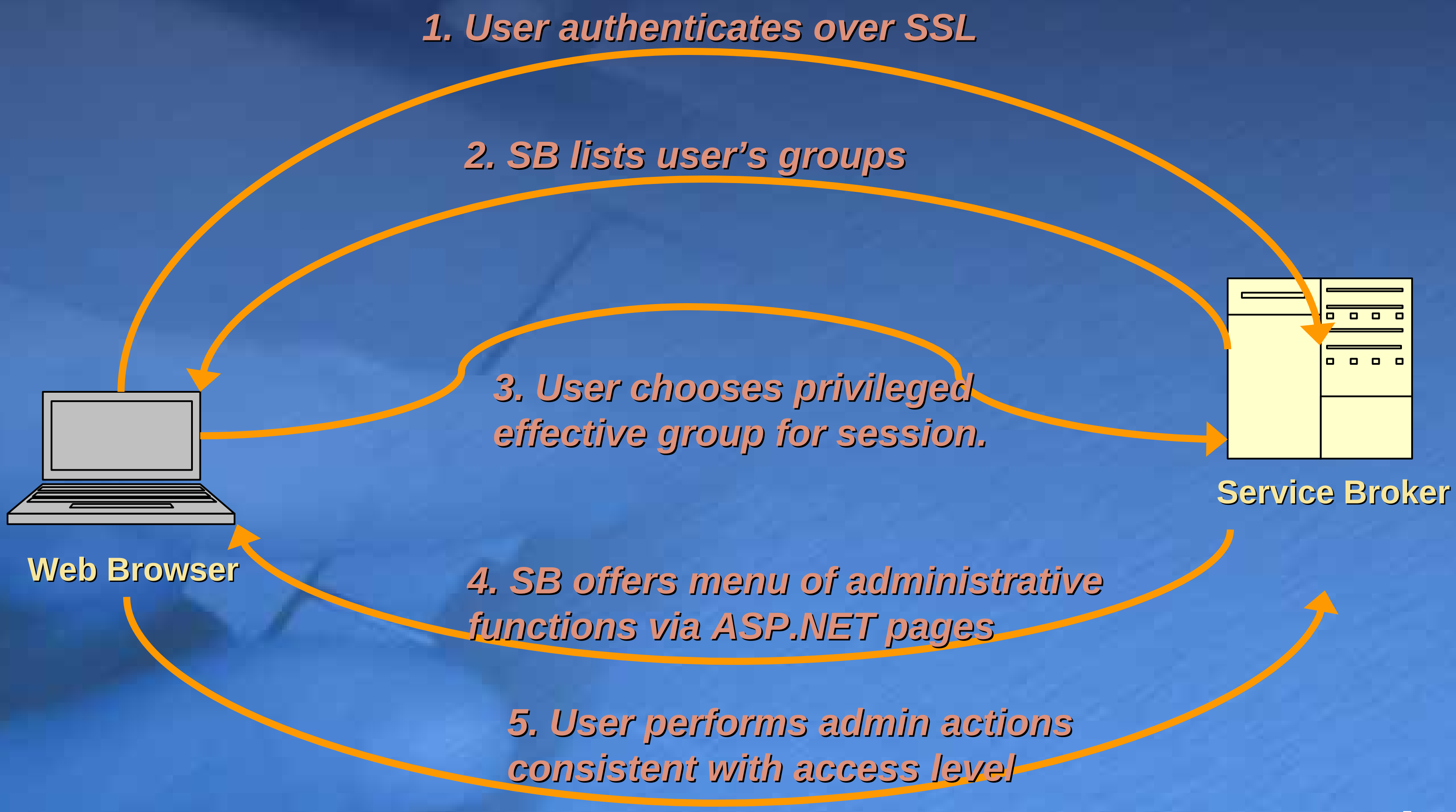
2 RetrieveResult(experimentID)

3 *returns* ResultReport

4 RetrieveResult(experimentID)

5 *returns* ResultReport

Administrator Web Session



1. User authenticates over SSL
2. SB lists user’s groups
3. User chooses privileged effective group for session.
Web Browser
Service Broker
4. SB offers menu of administrative functions via ASP.NET pages
5. User performs admin actions consistent with access level

Service Broker Administrative Services

- Adding, modifying, and removing lab servers and clients.
- Adding, removing, or confirming user access.
- User management including assigning users to groups and modifying access rights.
- Managing experiment records.

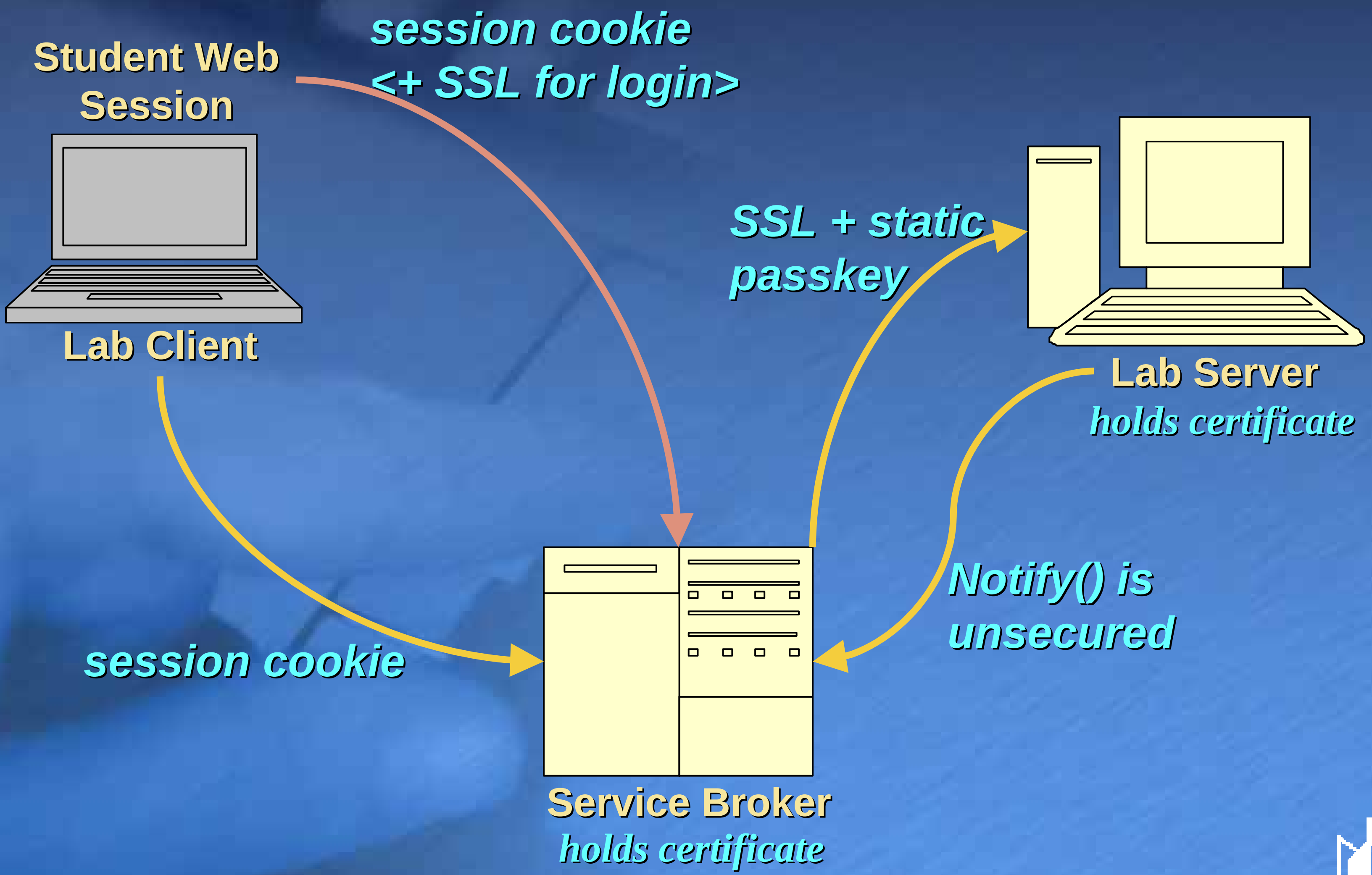
iLab Authentication

- The iLab Service Broker provides a default implementation of a basic user name and authentication scheme.
- The system architecture and data model allows for alternate authentication mechanisms, e.g., Kerberos or client certificates, but we have not implemented an example.

iLab Authorization

- iLab users are assigned to *groups*, most of which correspond to courses which have access to labs.
- Once the Service Broker has identified a user, it allows the user to choose his or her *effective group* for the session.
- The effective group corresponds to a *role* or *credential set* with an associated list of permissions (*grants* in iLab terminology).
- The *superuser group* gives its members all permissions when it is chosen as the effective group for an administrative session.
- Each user has default permission to read and write documents such as experiment records that they create.

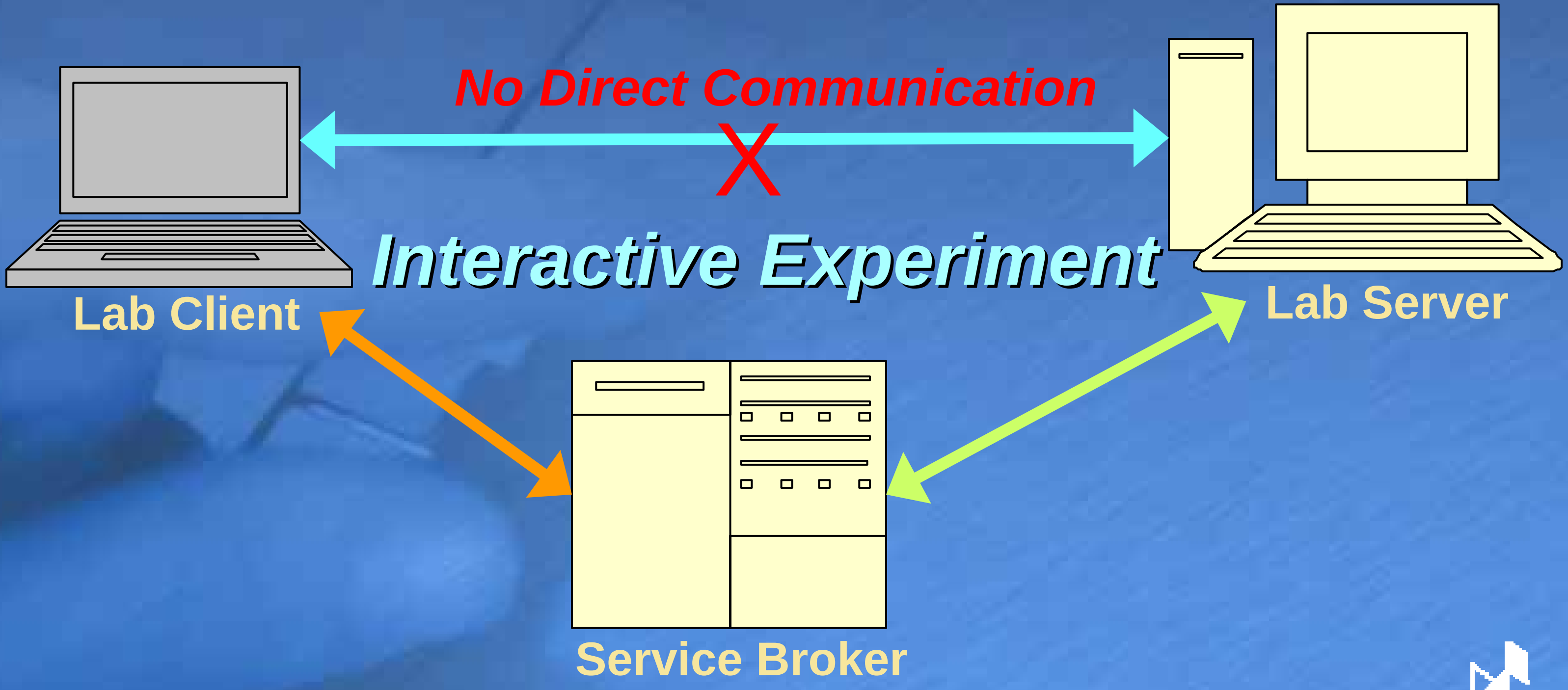
iLab Security



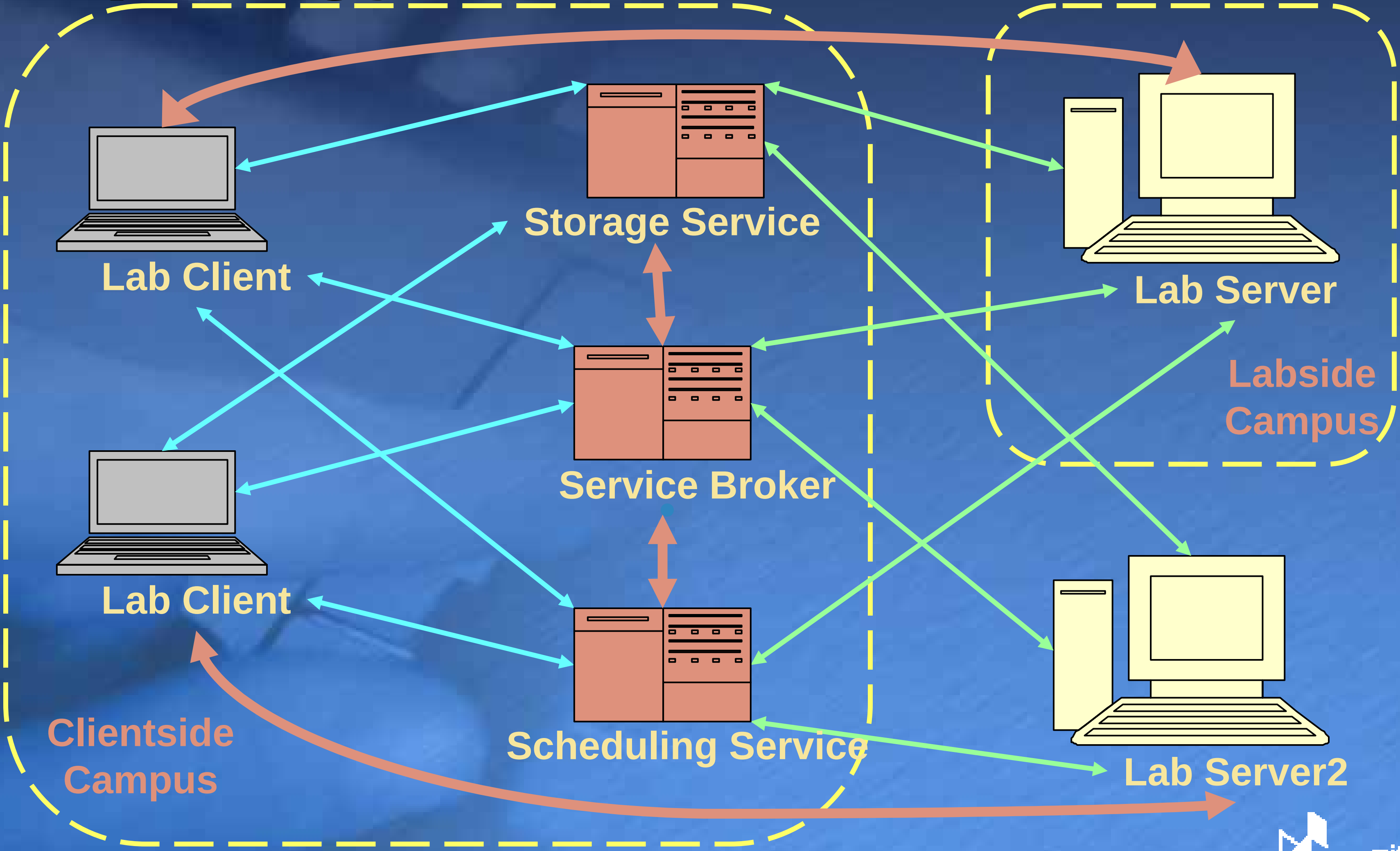
26 June 2005

Batched Experiment Network Topology

In the batched experiment architecture, the client and the lab server communicate only through the Service Broker:



Preliminary Interactive Topology

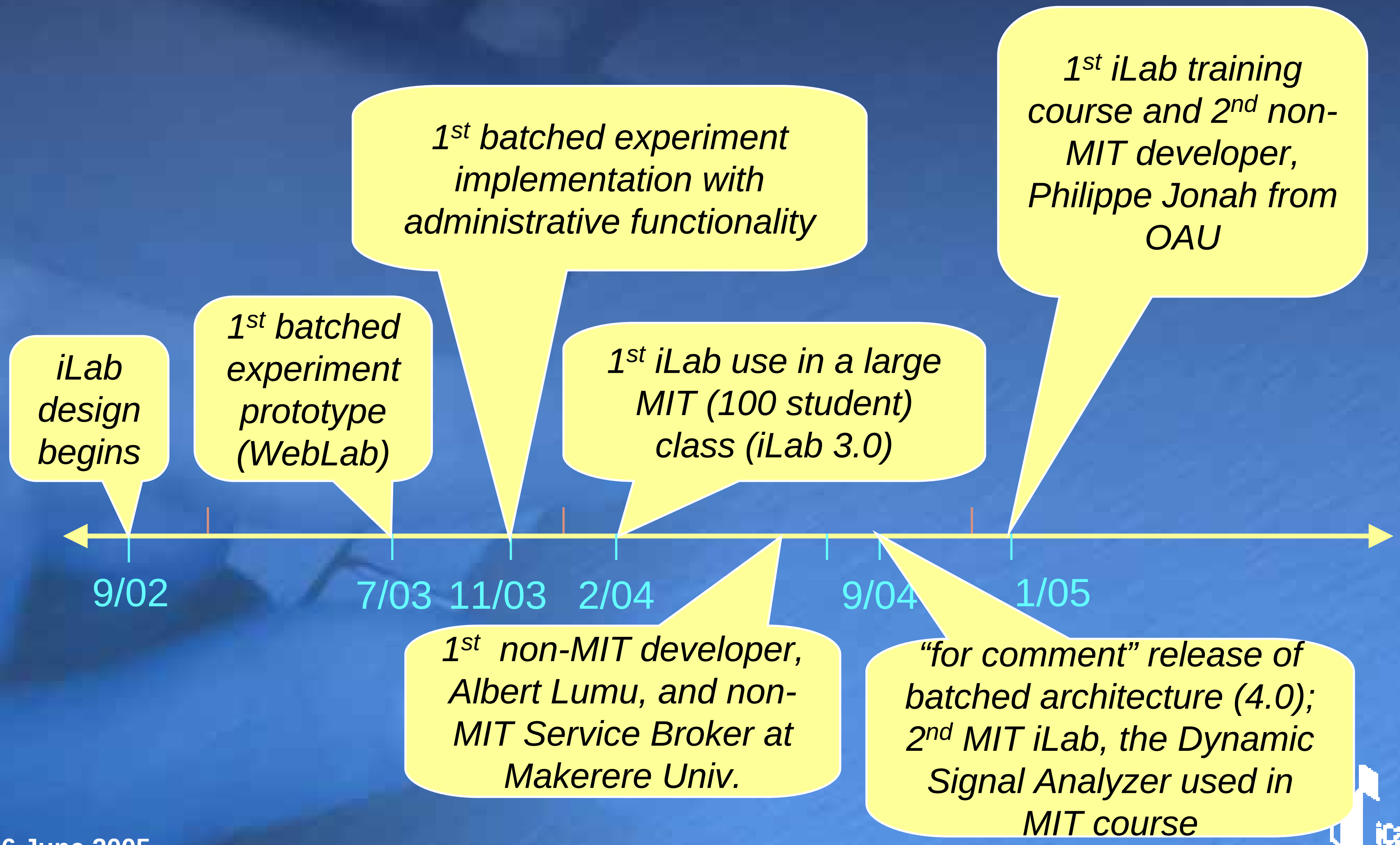


26 June 2005

mit iCampus

iLab Shared Architecture:

Project Timeline, 1



9/02

iLab design begins

7/03

1st batched experiment prototype (WebLab)

11/03

1st batched experiment implementation with administrative functionality

2/04

1st iLab use in a large MIT (100 student) class (iLab 3.0)

1st non-MIT developer, Albert Lumu, and non-MIT Service Broker at Makerere Univ.

9/04

"for comment" release of batched architecture (4.0); 2nd MIT iLab, the Dynamic Signal Analyzer used in MIT course

1/05

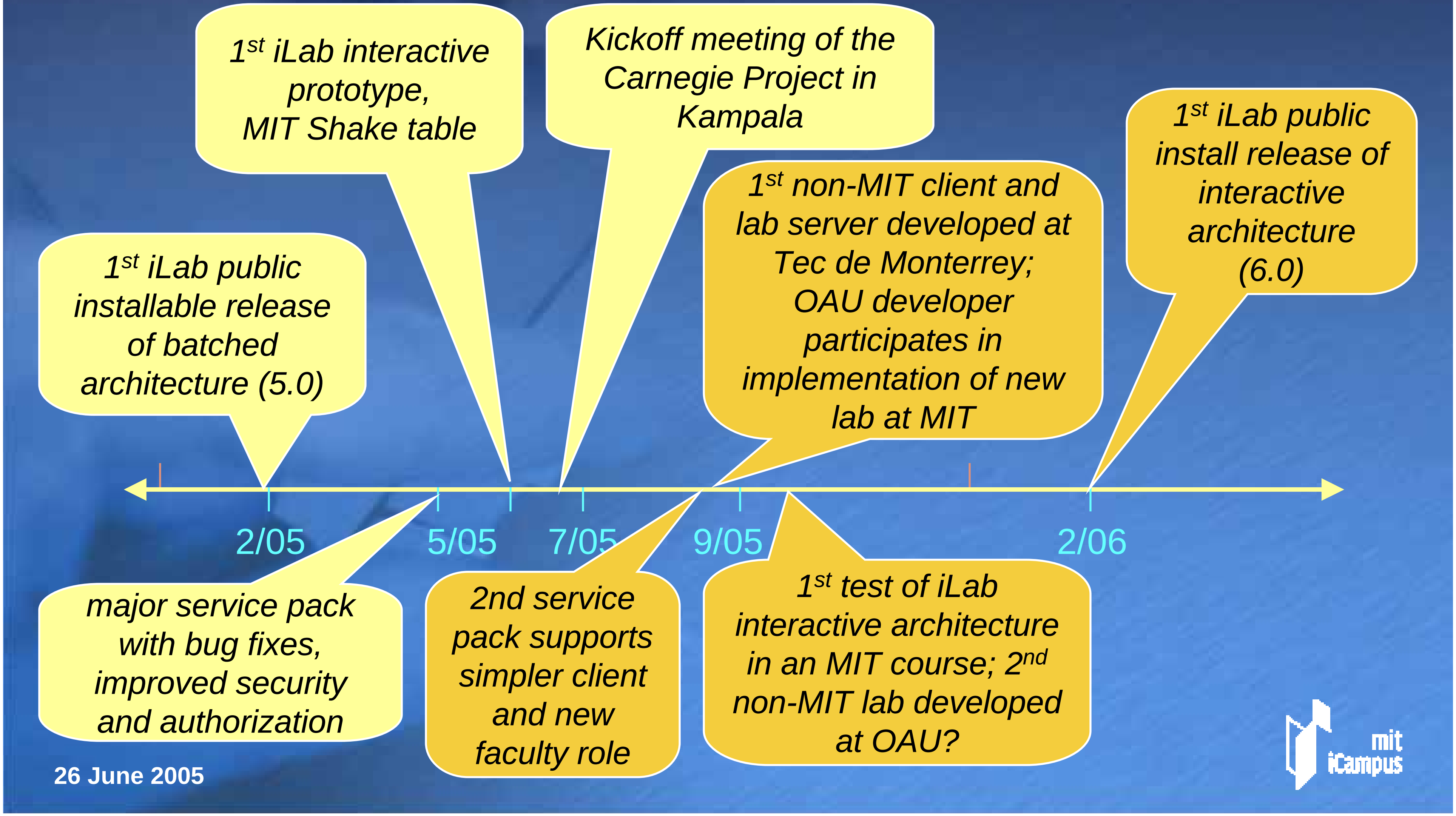
1st iLab training course and 2nd non-MIT developer, Philippe Jonah from OAU

Lab deployment through iLab Shared Architecture

- Microelectronics WebLab 6.0:
  - Developed by Jim Hardison and David Zych
  - Deployed Feb. 2004 in MIT undergrad course
  - Main System since Fall 2004
- Dynamic Signal Analyzer:
  - Developed from scratch in 9 months by Gerardo Viedma and Kent Lundberg
  - Deployed Sept. 2004 in MIT undergraduate subject

iLab Shared Architecture:

Project Timeline, 2



1st iLab public installable release of batched architecture (5.0)

major service pack with bug fixes, improved security and authorization

1st iLab interactive prototype, MIT Shake table

Kickoff meeting of the Carnegie Project in Kampala

2nd service pack supports simpler client and new faculty role

1st non-MIT client and lab server developed at Tec de Monterrey; OAU developer participates in implementation of new lab at MIT

1st test of iLab interactive architecture in an MIT course; 2nd non-MIT lab developed at OAU?

1st iLab public install release of interactive architecture (6.0)

26 June 2005

Shaketable Prototype

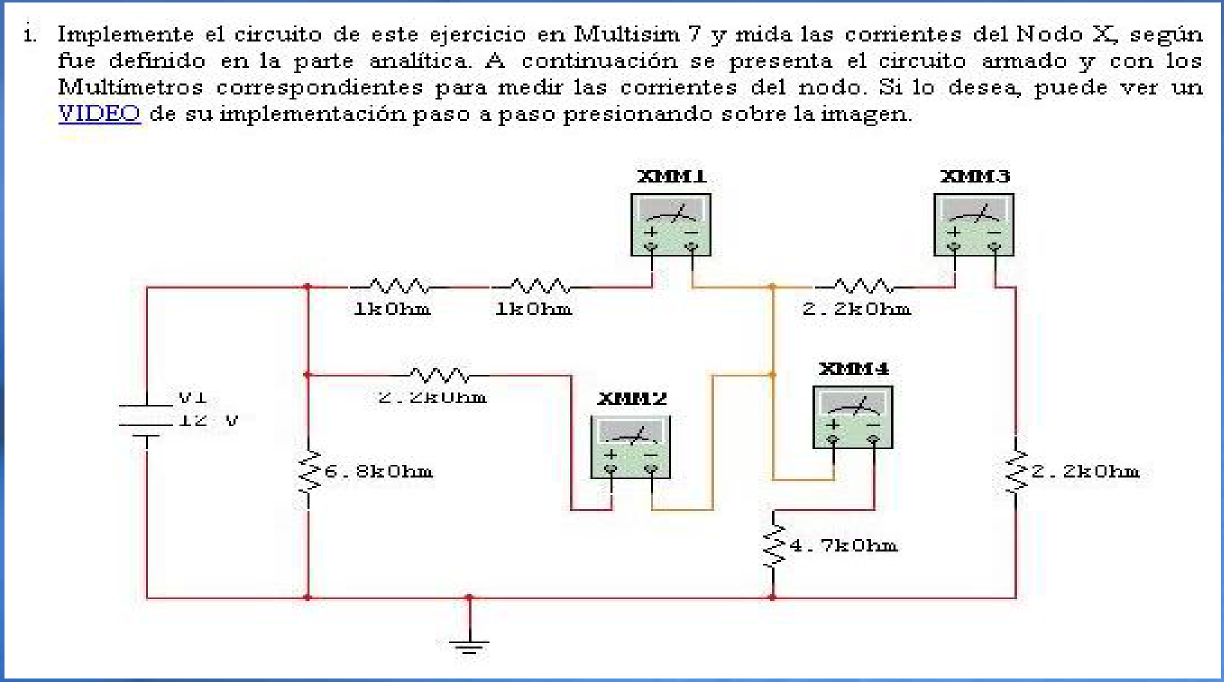
Major Milestone:

The 1st iLab Interactive Lab

- Uses the new iLab interactive authorization (ticket) architecture
- Does not disrupt the original implementation

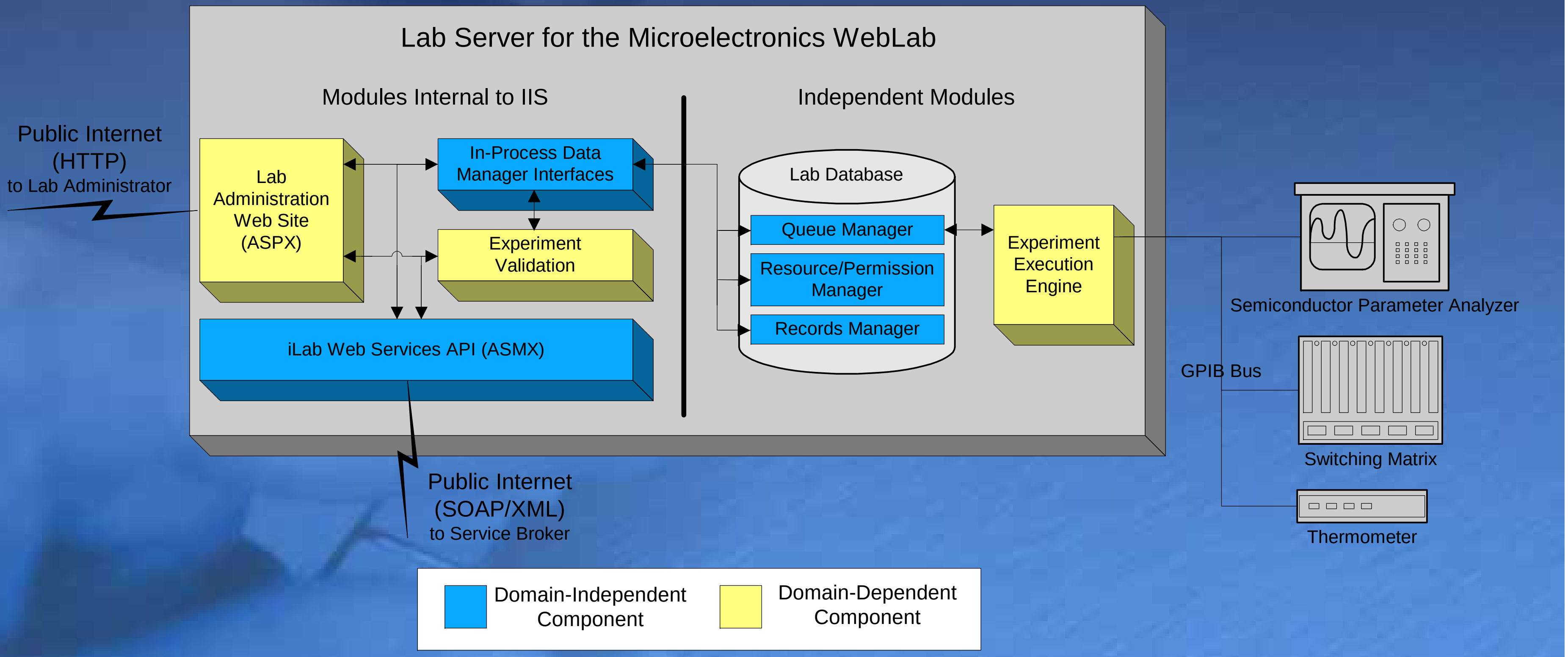
Collaboration with Tec de Monterrey, Summer 2005

- Development of the 1st non-MIT iLab based on current web-enabled experiments:



i. Implemente el circuito de este ejercicio en Multisim 7 y mida las corrientes del Nodo X, según fue definido en la parte analítica. A continuación se presenta el circuito armado y con los Multímetros correspondientes para medir las corrientes del nodo. Si lo desea, puede ver un VIDEO de su implementación paso a paso presionando sobre la imagen.

The WebLab 6.0 Lab Server



Lab Server for the Microelectronics WebLab

Modules Internal to IIS

Independent Modules

Public Internet
(HTTP)
to Lab Administrator

Lab Administration Web Site (ASPX)

In-Process Data Manager Interfaces

Experiment Validation

iLab Web Services API (ASMX)

Lab Database

Queue Manager

Resource/Permission Manager

Records Manager

Experiment Execution Engine

Semiconductor Parameter Analyzer

GPIB Bus

Switching Matrix

Thermometer

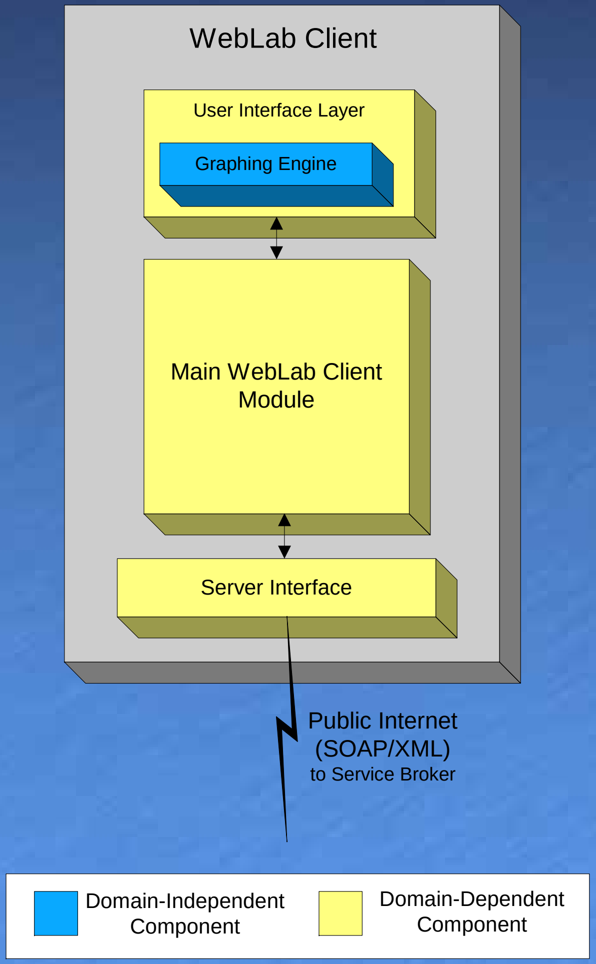
Public Internet
(SOAP/XML)
to Service Broker

🟦 Domain-Independent Component

🟨 Domain-Dependent Component

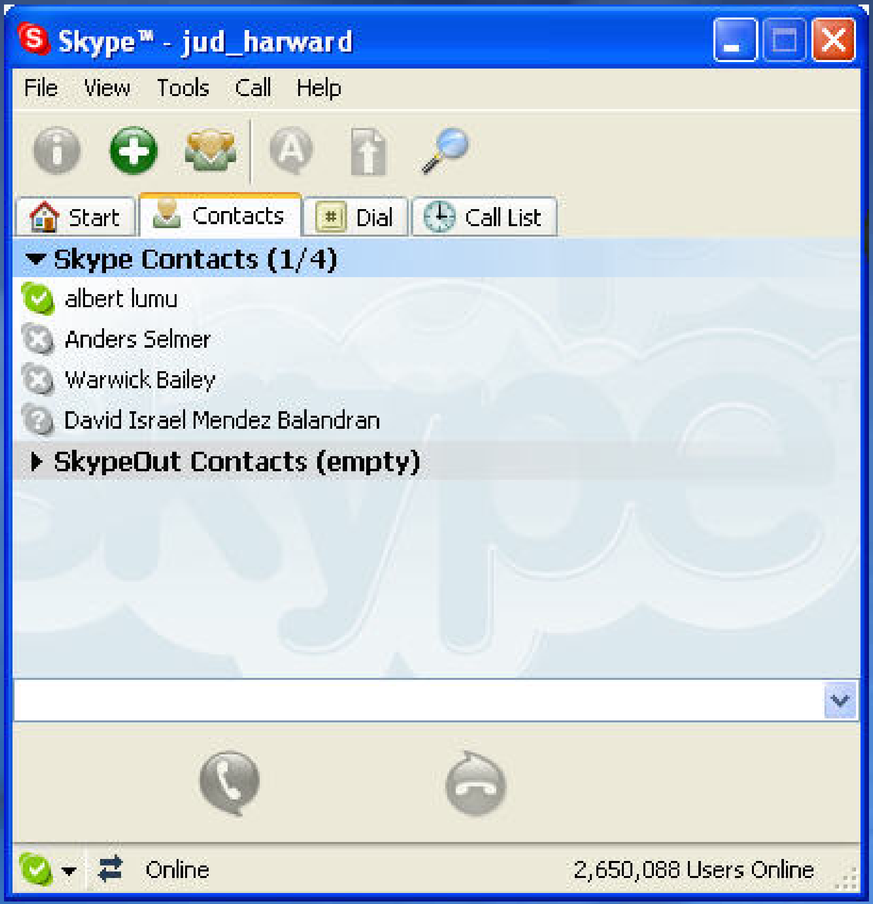
The WebLab 6.0 Client

- Three components:
  - User Interface Layer
  - WebLab Client Core Module
  - Server Interface
- Most client code is lab-specific.



26 June 2005

iLab Partners Developer Support



▼ Skype Contacts (1/4)

- albert lumu
- Anders Selmer
- Warwick Bailey
- David Israel Mendez Balandran

▶ SkypeOut Contacts (empty)

Online — 2,650,088 Users Online

- Developer visits
- Release of standard lab server and client modules
- VoIP conferencing
  - world-wide virtual development team

iLab Intellectual Property Policy

- All MIT developed software has been and will continue to be made available for free under an open source license.
- We encourage but do not require our academic partners to follow the same policy. The decision to share their code and under what terms is their to determine.
- We allow industrial partners to develop commercial “shrink-wrapped” (supported) versions of the iLab components.