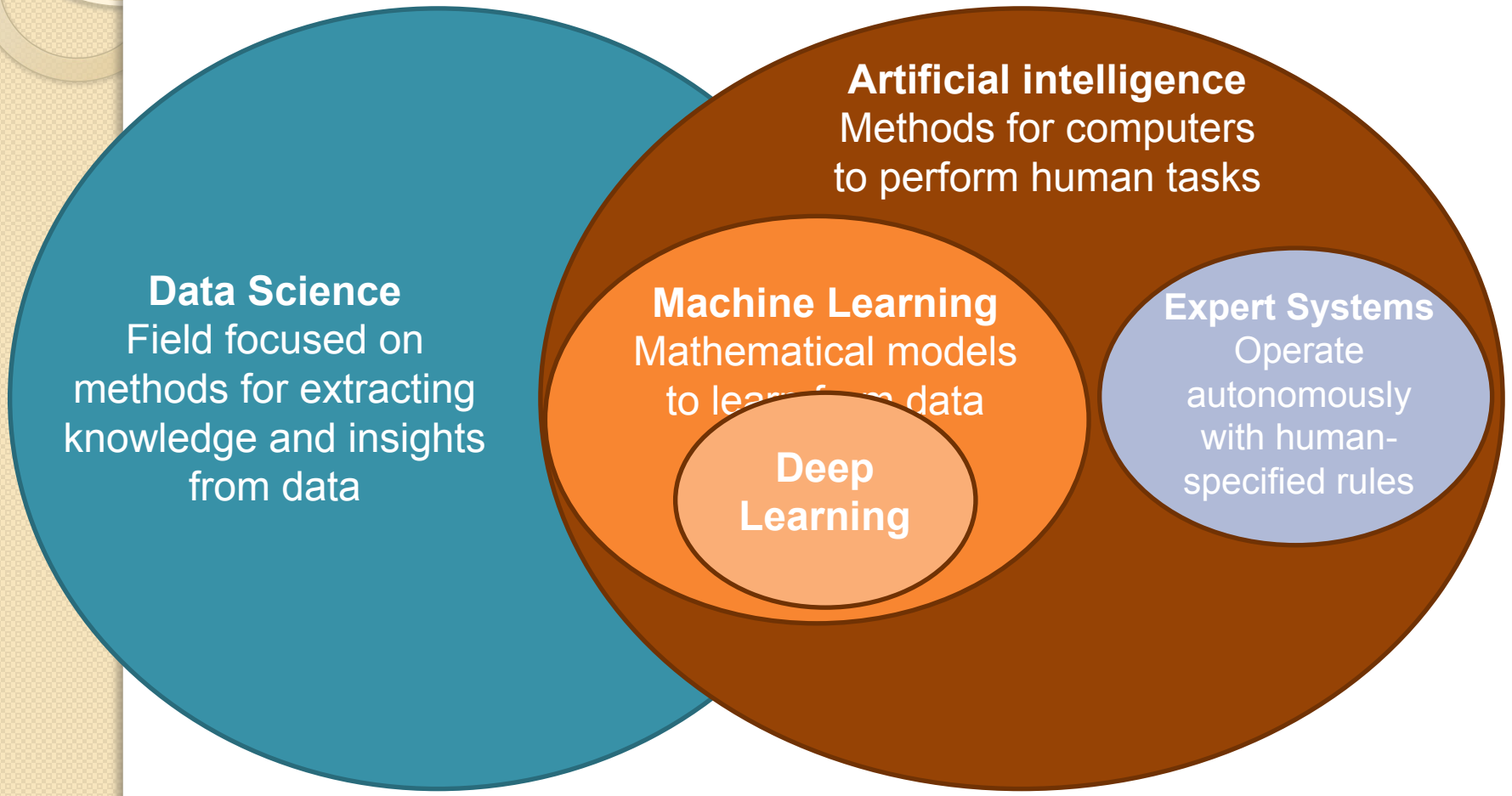
DSCI-560
Lecture 3: Data Science in Practice
Data Science Professional Practicum

Young Cho
Department of Electrical Engineering
University of Southern California

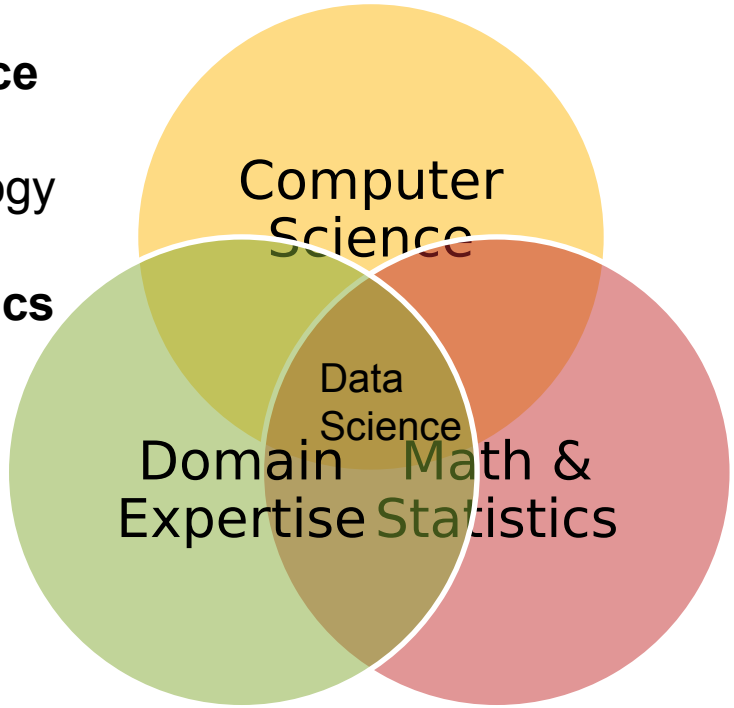
What is Data Science



Multi-disciplinary Skills

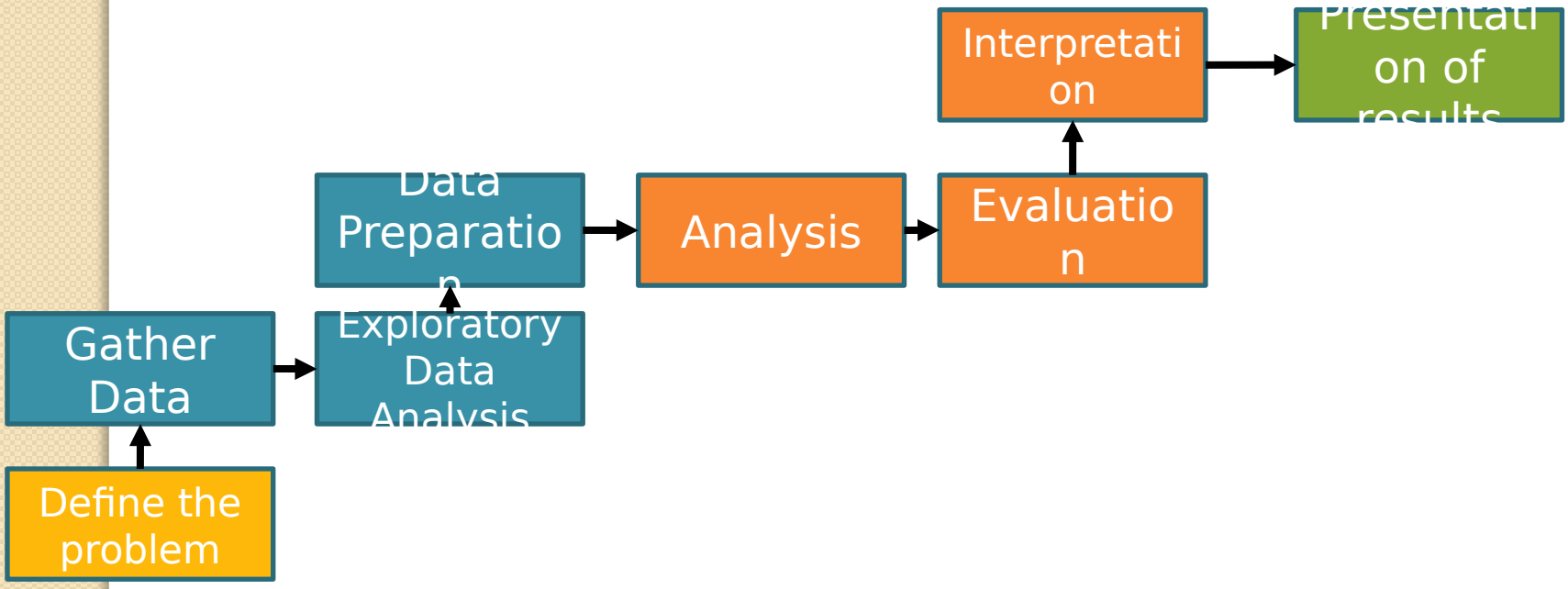
Computer Science
Programming
Big Data Technology

Math and Statistics
Machine Learning
Multivariate
Calculus/Algebra



Domain Expertise
Expert systems
UI/UX
Visualization

Data Science Pipeline

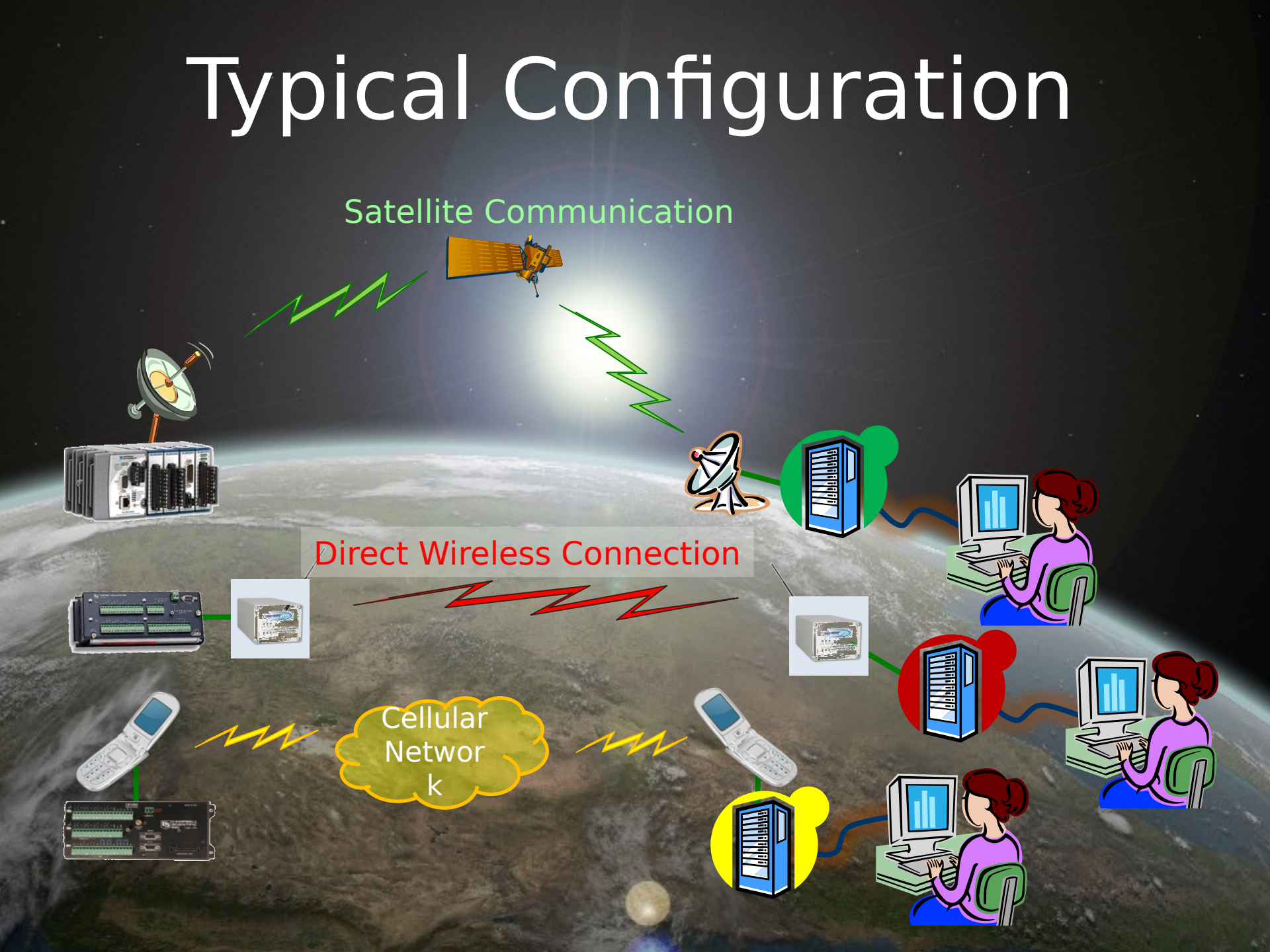


Gathering Data

Example Project: Satellite Sensor Gateway (2008-2009)

- Synergistic efforts at USC/ISI to develop several systems
  - Satellite Sensornet Gateway
  - Costa Rica international cyberinfrastructure
  - USC/ISI – CENS SensorKit
  - National Ecological Observatory Network (NEON)
  - Global Lake Ecological Observatory Network (GLEON)
- Relevant to Scientific Applications
  - Feedback from the Science Advisory Board members
  - Closely collaborate with scientists
- Deployment Driven Design
  - Build, Test, and Revise

Typical Configuration



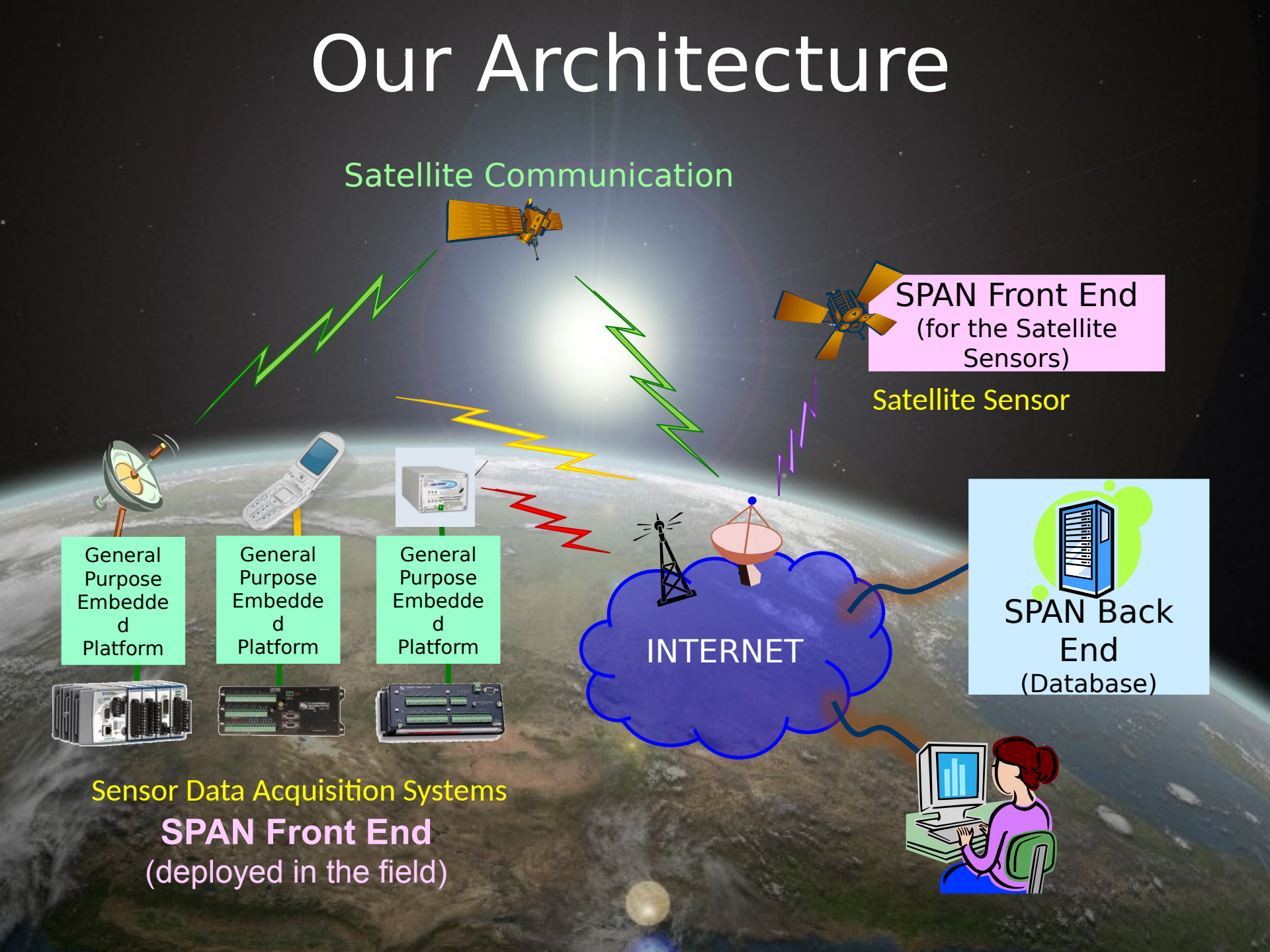
Real-World Issues

- Hardware Dependency
  - Use components from the same company or their partners
  - Software closely integrated with the hardware
  - Difficult to program
  - Application-specific
- Proprietary/Domain-Specific Standards
  - Incompatible standard from one system to another
  - Proprietary communication standard
- Proprietary Back-End Applications
  - Constrained to use proprietary back-end
  - Sharing possible at the highest level

Deeper Understanding

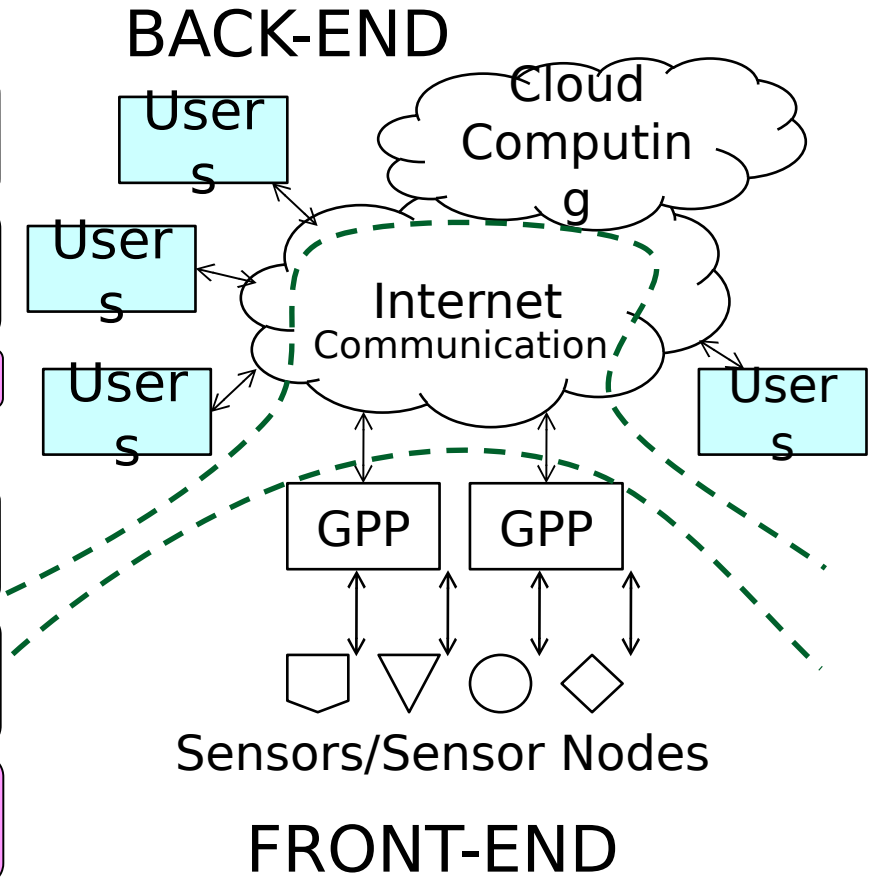
- Experience
  - One solution does not fit every case
  - Different environments require different tools
  - Lower cost and lower power is desirable
- Observations
  - Scientists use simple functionalities (raw data and excel sheets)
  - Not traditional wireless sensor network nodes (wireless sensors)
  - Very difficult to collaborate (proprietary and custom designs)
  - Simplest is the best (lowest power and cost)

Our Architecture



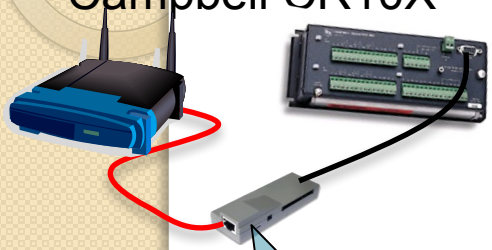
Architecture

- Hardware
  - Variety of Data Acquisition Units
  - General Purpose Processing Platform
  - Variety of Physical Communication
  - Internet Infrastructure
  - Scalable Servers on the Web
- Software
  - Drivers - Application Programming Interface
  - User Specified Applications
  - Communication Protocols
  - Markup-Languages
  - Cloud Computing Resources (i.e. Google)
  - Scalable Database



Front-end Configurations

Campbell CR10X



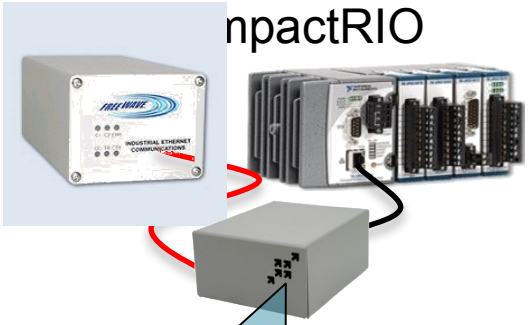
802.11g

Triggered dynamic reconfiguration policies

Command translation for Campbell dataloggers

CR10X data format, CR10X Rec/Replay, Serial Interface

npactRIO



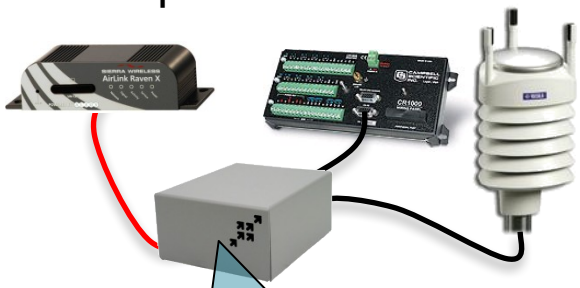
900MHz Freewave radio

Triggered dynamic reconfiguration policies

Command translation for compact RIO dataloggers

cRIO data format, ASCII commands over Ethernet

Campbell CR1000



3G Cellular Network

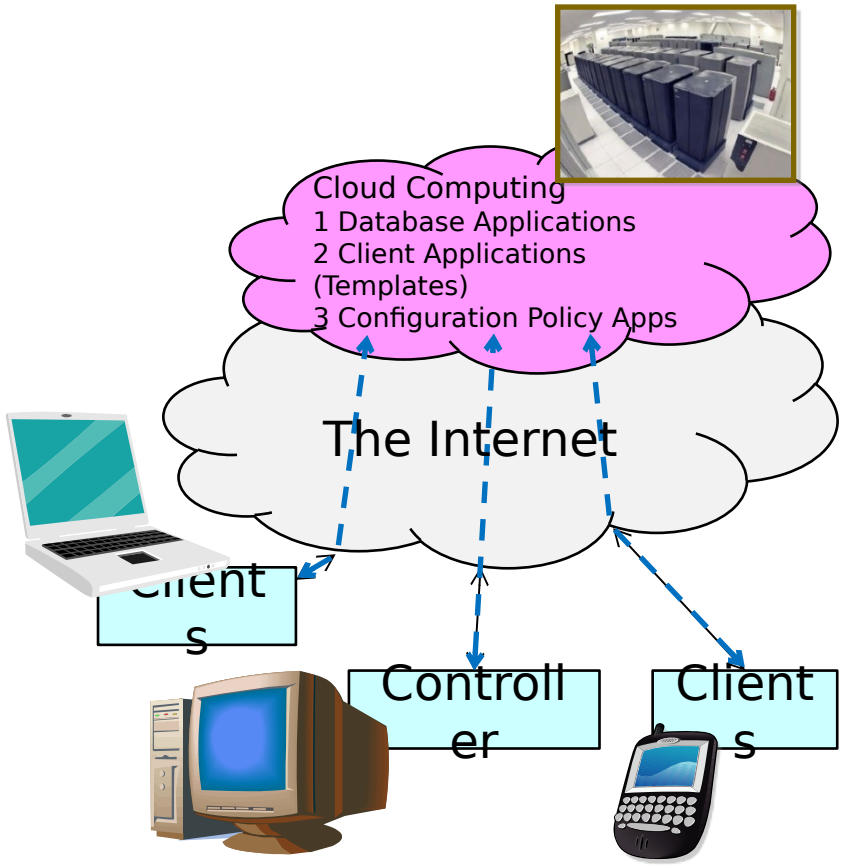
Triggered dynamic reconfiguration policies

Command translation for CR1000 & vaisalar weatherstation

CR1000 data format, PakBUS over TCP/IP, Interface for Vaisalar

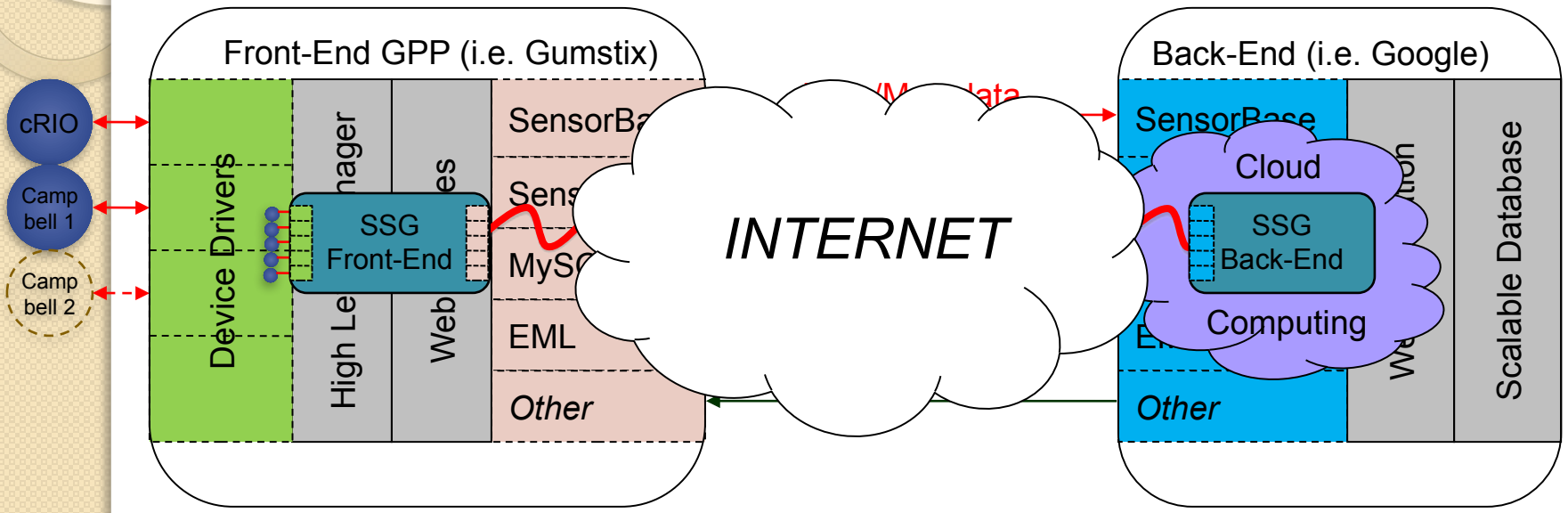
Internet and Cloud Computer

- Network Communication
  - Consumer Satellite
  - Cellular Network
  - Long Range Modems
  - 802.11x Wireless
  - **The Internet**
- Cloud Computing System
  - Specialized Distributed Web/DBASE
  - Highly Scalable and Accessible
  - Professionally Maintained
  - Updated with the Technology Trend
  - Free or Low fee subscription
  - **Google Application Engine, Amazon and**



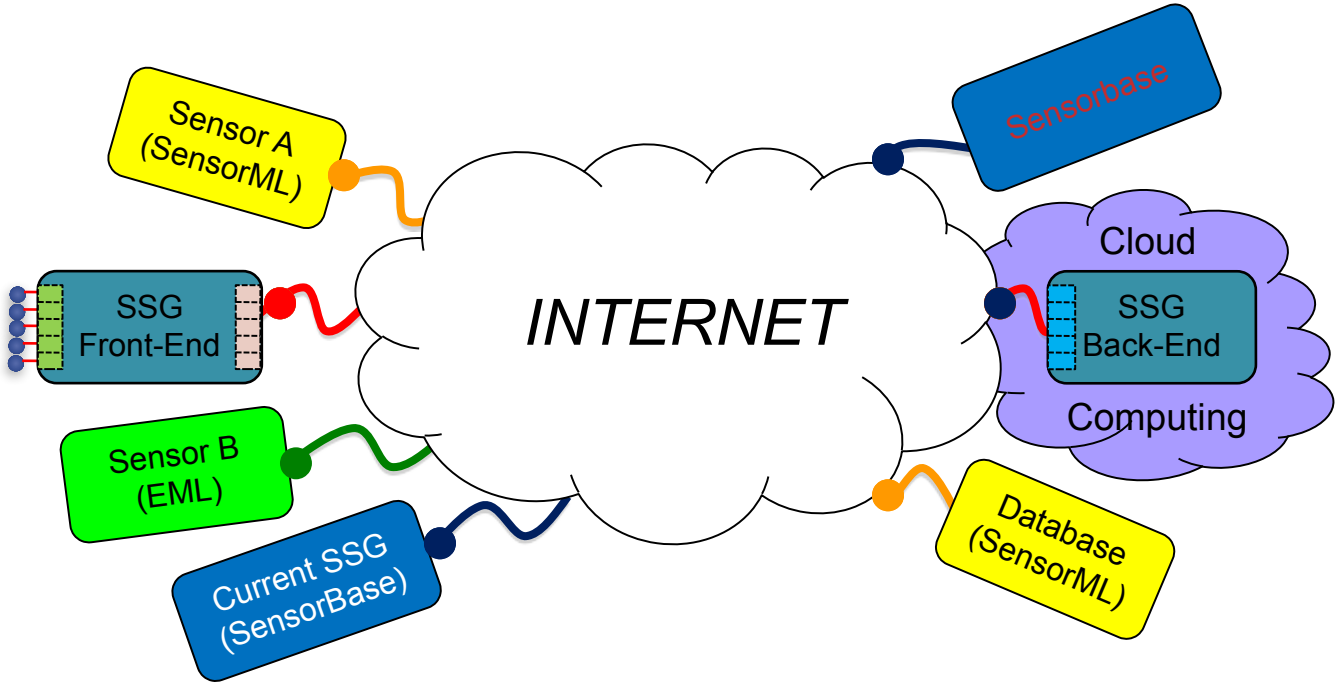
Device with Web-browsing Capability

Universal Data Acquisition Interface



- **Programmable Generic Interface**
  - Simple but powerful scripting language to parse data
  - Ability to support many markup-languages (ML)
- **Plug-in Interface Module for Each ML**
  - SensorBase Commands and Data Packets
  - Extended ML (XML), SensorML, Environmental ML (EML), and others

Universal Data Acquisition Interface

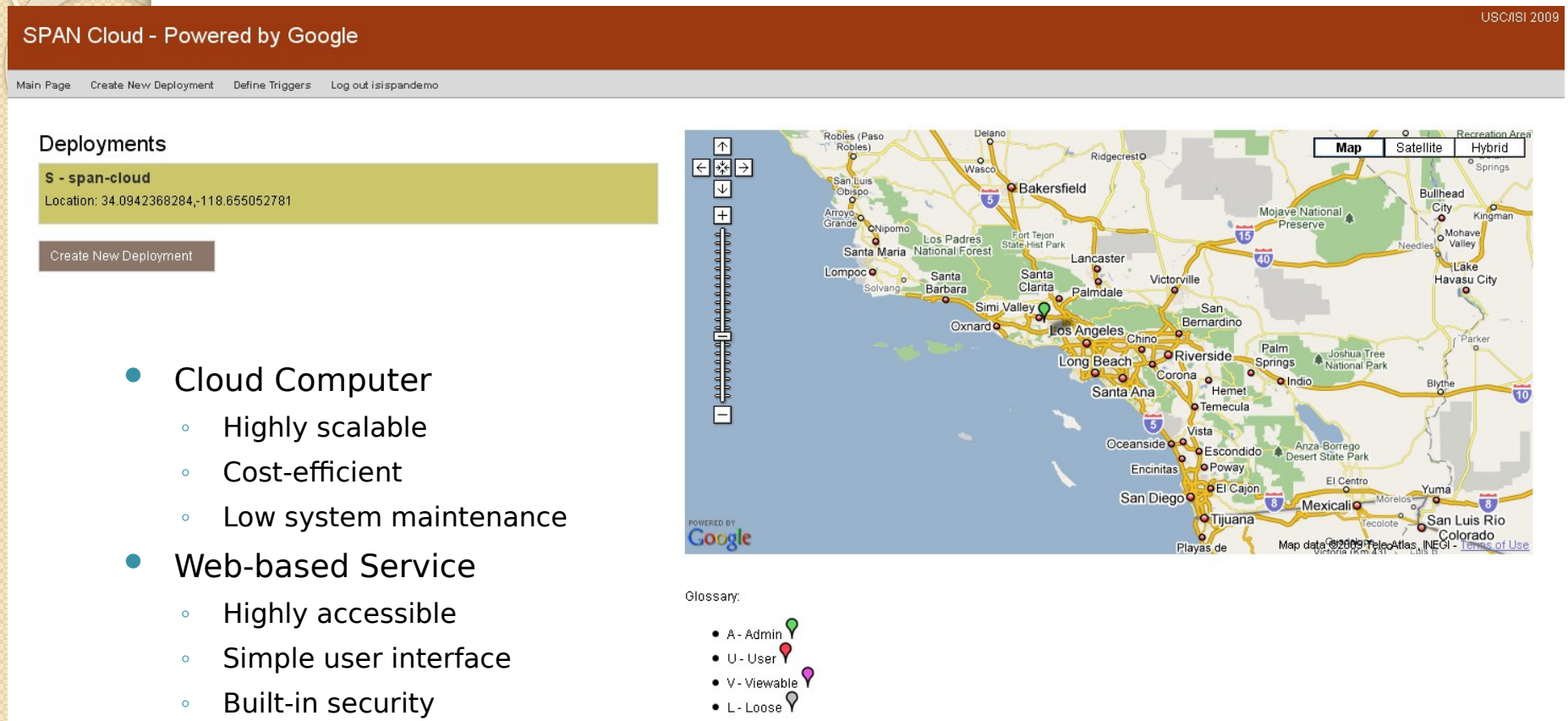


- **Programmable Generic Interface**
  - Simple but powerful scripting language to parse data
  - Ability to support many markup-languages (ML)
- **Plug-in Interface Module for Each ML**
  - SensorBase Commands and Data Packets
  - Extended ML (XML), SensorML, Environmental ML (EML), and others

SPAN Front-end Implementation

- Device driver support for widely used Sensors and Dataloggers
  - UC Berkeley Motes
  - National Instruments Compact RIO
  - Campbell Scientific dataloggers (legacy and PAKBUS based)
  - Vaisala weather station
- Device driver support for variety of communication channels
  - Satellite, cellular, WiFi, and various others with Ethernet interface
  - Freewave long range radio with serial interface
- Generalized data handling
  - Data cache and reliable transport
  - Standardized data format and communication protocols
- Revision and port of SPAN for generic embedded processor
  - Port of generic command and control functions
  - Dynamic triggering capabilities

SPAN Cloud – Database on Google Cloud



- Cloud Computer
  - Highly scalable
  - Cost-efficient
  - Low system maintenance
- Web-based Service
  - Highly accessible
  - Simple user interface
  - Built-in security
  - Easily deployable

SPAN Cloud - Visual of Sensors and Activities

SPAN Cloud - Powered by Google

USC/ISI 2009

Main Page | span-cloud info page | Add New Sensor

span-cloud

Latitude (e.g. 34.0412432)

34.0942368284

Longitude (e.g. -120.1182321)

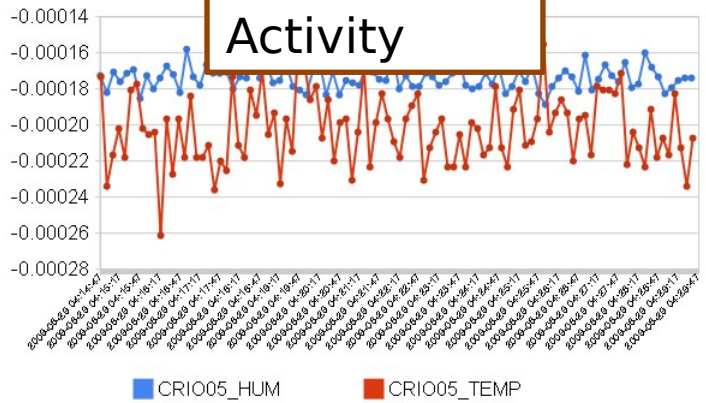
-118.655052781

Comments

Modify Information

A view of the deployment

Recent Activity



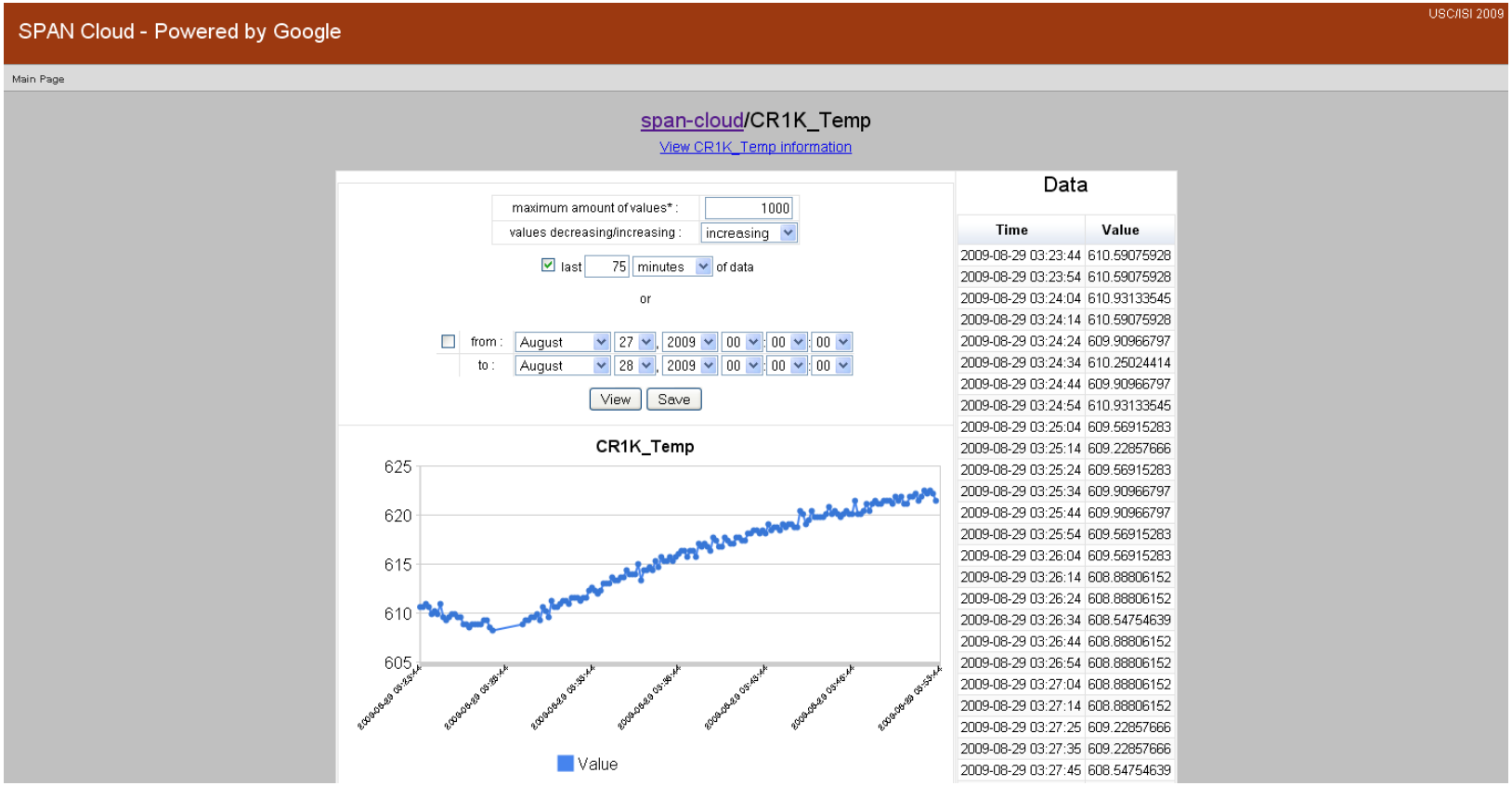
- View span-cloud data page

Sensors in span-cloud

CRIO05_HUM
CRIO05_TEMP
CRIO05_SYSLOG
CR1K_Batt_Volt
CR1K_Temp
CR1K_Humidity
CR1K_Pressure
CR1K_PAR

Deployment Sensors

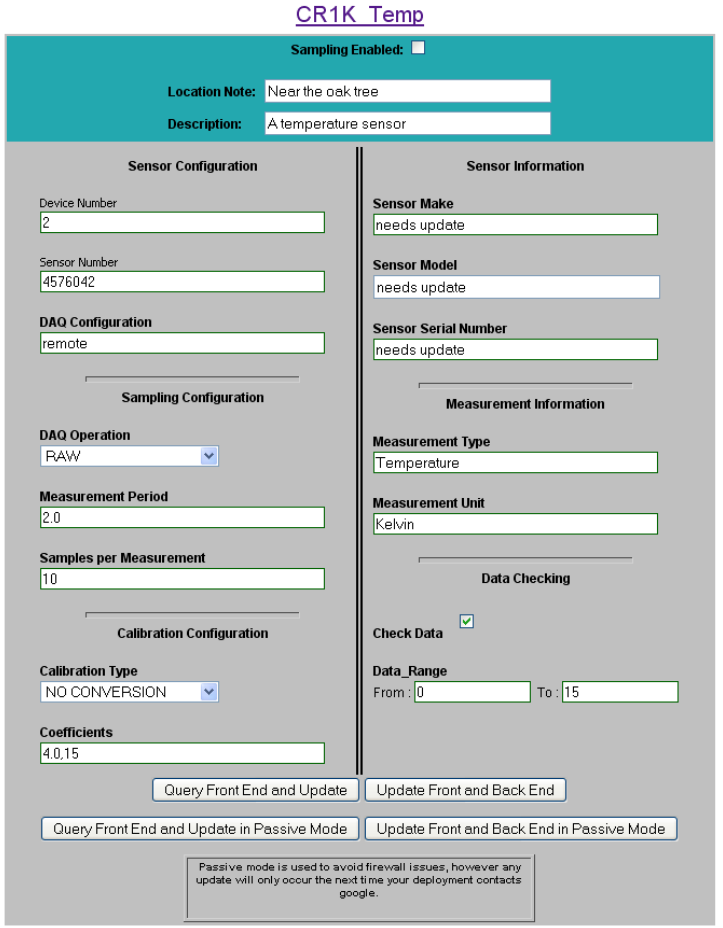
SPAN – Data Retrieval



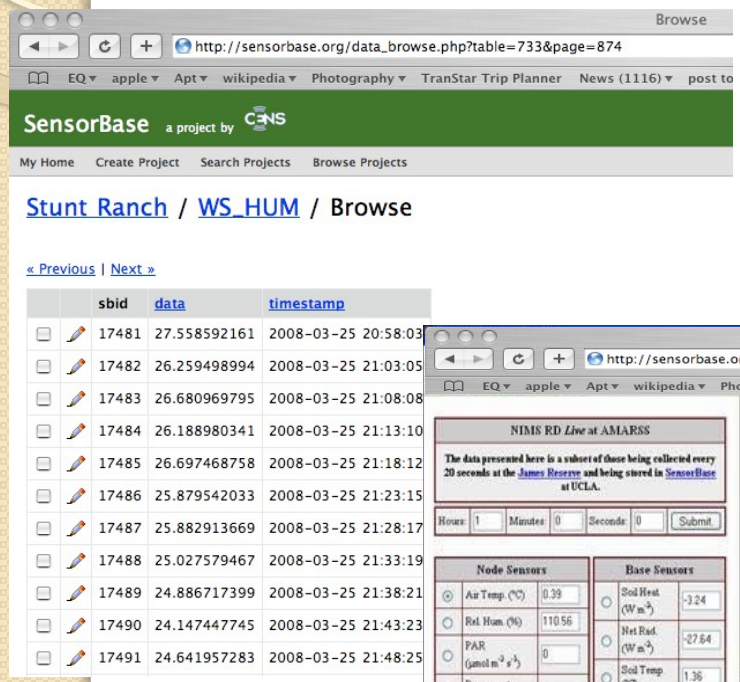
- Web-based Graphical and Text view of the data
- Exportable in Microsoft Excel compatible

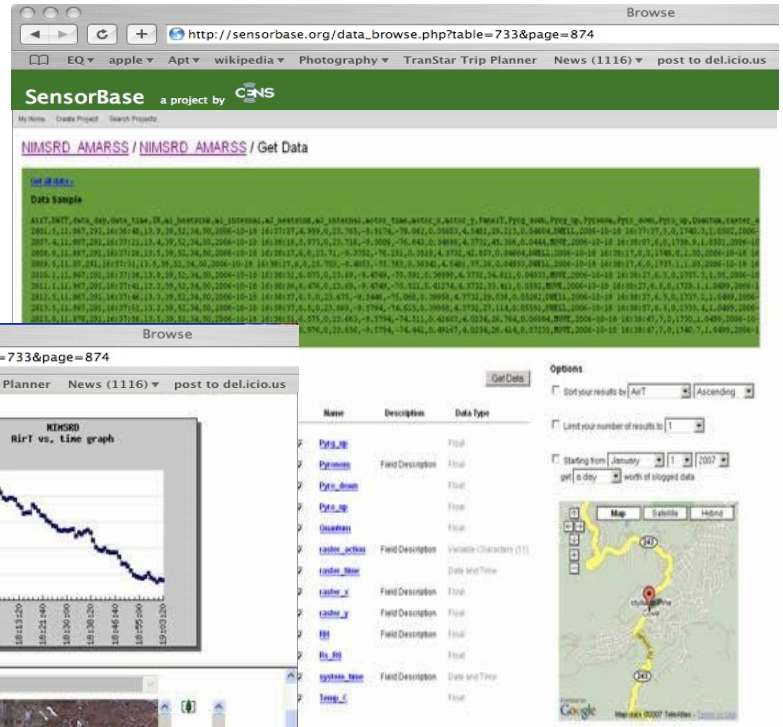
SPAN - Web-based Sensor (Re)Configurations

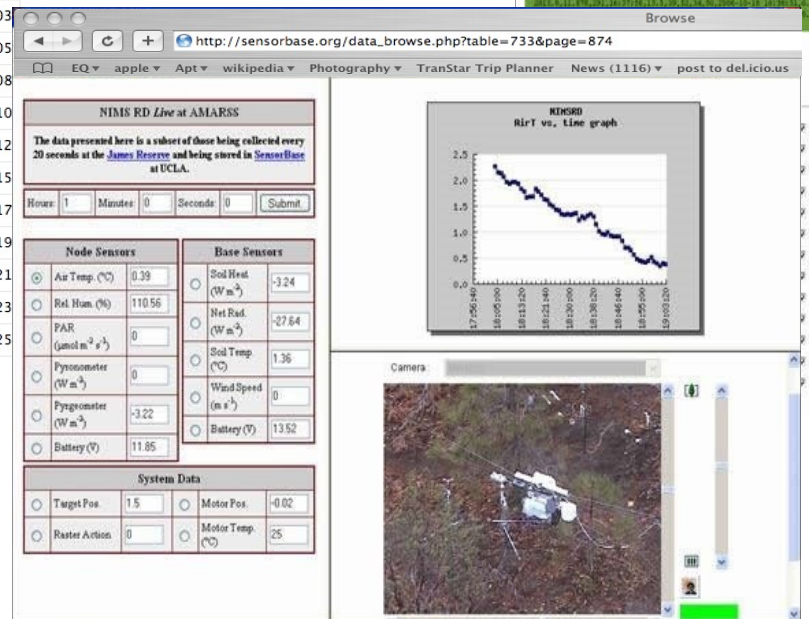
- Web-based Interface
  - Highly accessible interface
  - Remote configuration via Google
  - Issues commands to sensors
- Configuration Checking
  - Bounds and consistency check
  - Send legal commands to Sensors



SensorBase Database Compatible



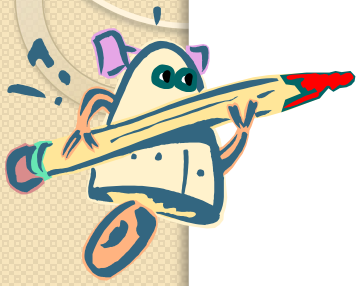




Data Preparation

- Example Project: Application Level Processor (2005-2008)
- Synergistic Effort between Gov. Agencies, University, and Companies
  - NSA and Air Force
  - SAIC and Aerospace Corporation
  - Washington University in St. Louis
- Data Quantity and Quality
  - Quantity needed to be intelligently reduced
  - Quality needed to be improved in terms of relevance
- Approach
  - Real-time Stream Data Preparation
  - Context Free Grammar based data cleanser
  - Hardware Accelerated system

Filtering with Pattern Matchers



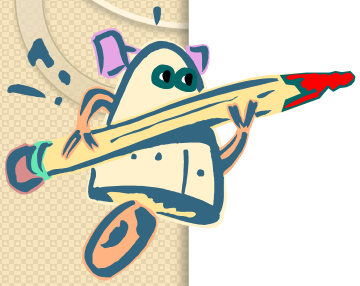
TAGBOT

HTML Source Document

<h2>Company Overview</h2>
 Corporate Fact Sheet Founded by <a href=
"/about/profile.html">Dr. J. Robert Beyster and a small group of scientists in 1969, SAIC, a Fortune 500 company, now ranks … and have more than 43,000 <!-- Also update employee number on: saic.com/news/0722.html --> employees with offices in over 150 cities.

- String Patterns
  - HTML tags can be detected and marked
  - Marks can be used to filter out the tags
  - Discrete gates, Memory based, Hybrid filter, Bloom filter and etc.
- Detect and Filter out HTML tags
  - <h2>, </h2>, , , <a href=, , <!--, -->
- Some unwanted texts are still not filtered away!

Beyond Pattern Matching



TAGBOT

HTML Source Document

```
<h2>Company Overview</h2>
<!-- Corporate Fact Sheet --> <p>Founded by <a href=
"/about/profile.html">Dr. J. Robert Beyster</a> and a
small group of scientists in 1969, SAIC, a Fortune 500
company, now ranks … and have more than 43,000 <!- Also
update employee number on: saic.com/news/0722.html -->
employees with offices in over 150 cities. </p>
```

Token List

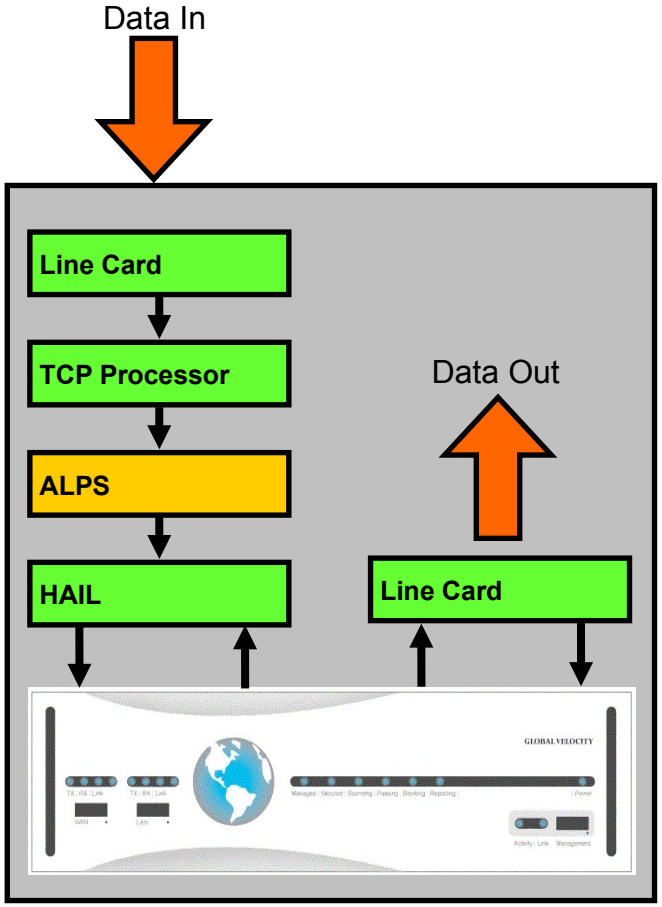
(1) hdr2 : ‘h2’
(2) para : ‘p’
(3) link : ‘a’
(4) href : ‘href=’
(5) quot : ‘”’.alpnum*.‘”’
(6) comm : alpnum*
(7) strg : alpnum*

Simple HTML Grammar

(1) Tag_Name ☾ hdr2 | para | link
(2) Comment ☾ ‘<!-’.comm.‘-->’
(3) Attrib ☾ href.quot | ε
(4) Tag_Head ☾ ‘<’.Tag_Name.Attrib.‘>’
(5) Tag_Tail ☾ ‘</’.Tag_Name.‘>’
(6) Expr ☾ Comment | strg | ε
(7) Line ☾ Tag_Head.Line.Tag_Tail
| Expr.Line.Expr | Expr
(8) Content ☾ Line.Content

Preprocessing TCP Flows

- Preprocess TCP flows
  - Remove unwanted information
    - Document tags
      - HTML, XML, Email, etc.
    - Non-text information
      - Executable files, email attachments, etc.
  - Pass remaining text data to HAIL
    - Help to improve the accuracy of HAIL
      - Remove junk data that may throw off HAIL
- Improve overall accuracy of the AFE
  - Classify based on only useful text data
- Process data at up to 10 Gbps



Grammar Specification

```
STRING [a-zA-Z0-9-]+
BITSTRING [0-1]+
%%
cards:      "<cards>" type "</cards>"
type:       config | cardplus
config:     "<config>" BITSTRING "</config>"
cardplus:   cardplus card | card
card:       "<card>" name title phone desc "</card>"
name:       "<name>" first last "</name>"
first:      "<first>" STRING "</first>"
last:       "<last>" STRING "</last>"
title:      "<title>" STRING "</title>"
phone:      "<phone>" STRING "</phone>
desc:       "<desc>" STRING "</desc>"
%%
```

Grammar Conversion

```
STRING [a-zA-Z0-9-]+
BITSTRING [0-1]+
%%
cards:      "<cards>" type "</cards>"
type:       config | cardplus
config:     "<config>" BITSTRING "</config>"
cardplus:   cardplus card | card
card:       "<card>" name title phone desc "</card>"
name:       "<name>" first last "</name>"
first:      "<first>" STRING "</first>"
last:       "<last>" STRING "</last>"
title:      "<title>" STRING "</title>"
phone:      "<phone>" STRING "</phone>
desc:       "<desc>" STRING "</desc>"
%%
```

Token List

0) <cards>	12) <last>
1) </cards>	13) [a-zA-Z0-9-]
2) <config>	14) </last>
3) [0-1]+	15) <title>
4) </config>	16) [a-zA-Z0-9-]
5) <card>	17) </title>
6) </card>	18) <phone>
7) <name>	19) [a-zA-Z0-9-]
8) </name>	20) </phone>
9) <first>	21) <desc>
10) [a-zA-Z0-9-]	22) [a-zA-Z0-9-]
11) </first>	23) </desc>

FIRST & *FOLLOW* Sets

Lexicon Core

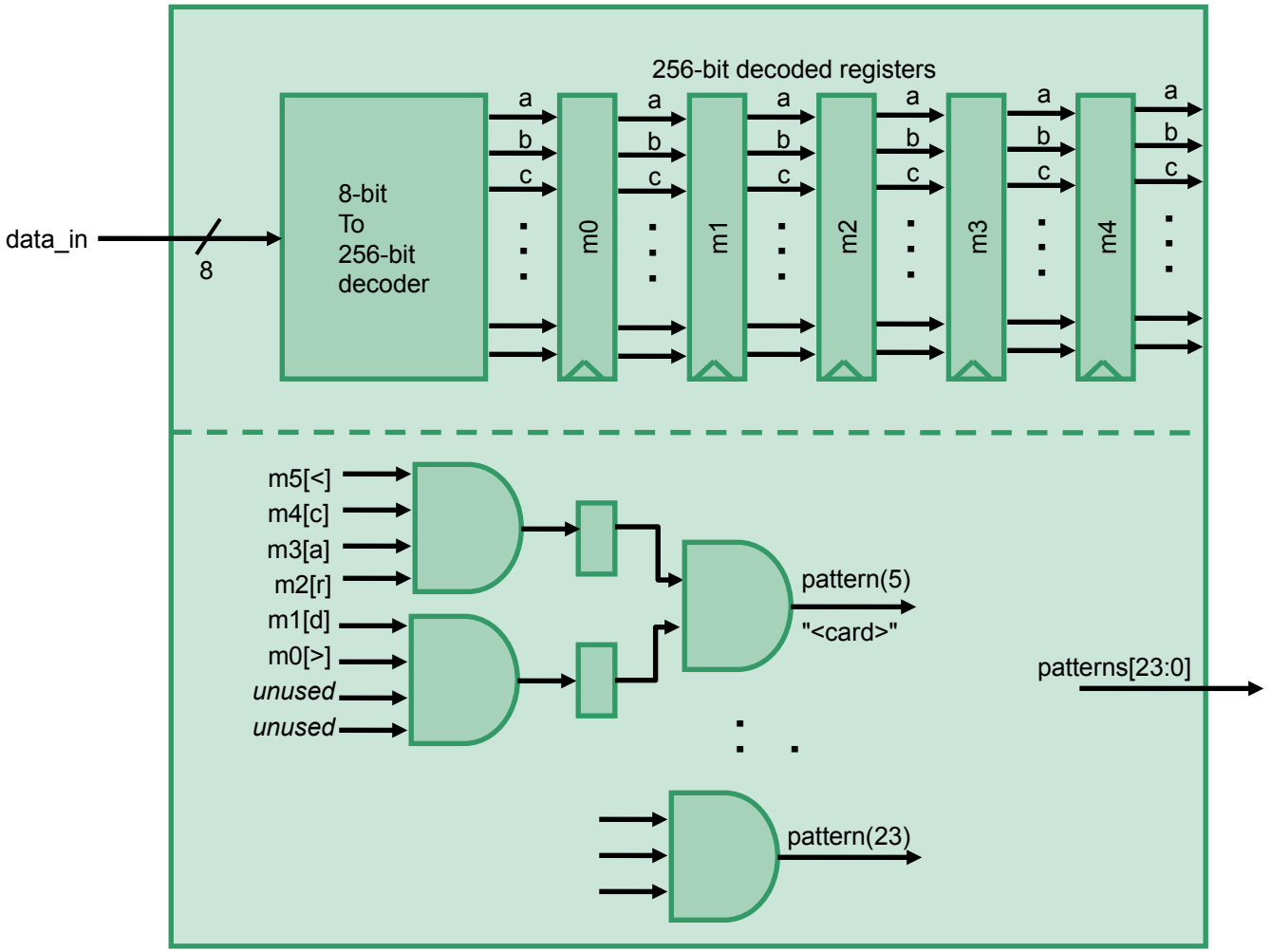
Parsing Structure

ALPS Core

Lexicon Core

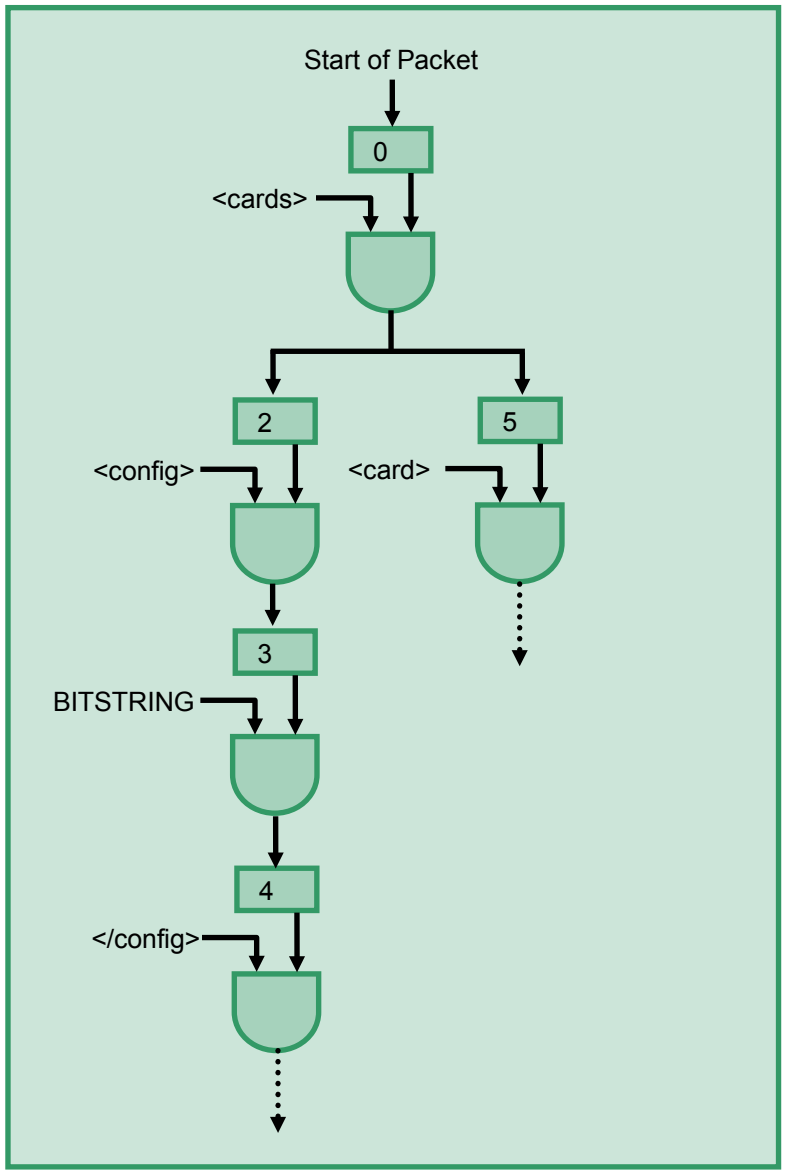
Token List

```
0)  <cards>
1)  </cards>
2)  <config>
3)  [0-1]+
4)  </config>
5)  <card>
6)  </card>
7)  <name>
8)  </name>
9)  <first>
10) [a-zA-Z0-9-]
11) </first>
12) <last>
13) [a-zA-Z0-9-]
14) </last>
15) <title>
16) [a-zA-Z0-9-]
17) </title>
18) <phone>
19) [a-zA-Z0-9-]
20) </phone>
21) <desc>
22) [a-zA-Z0-9-]
23) </desc>
```

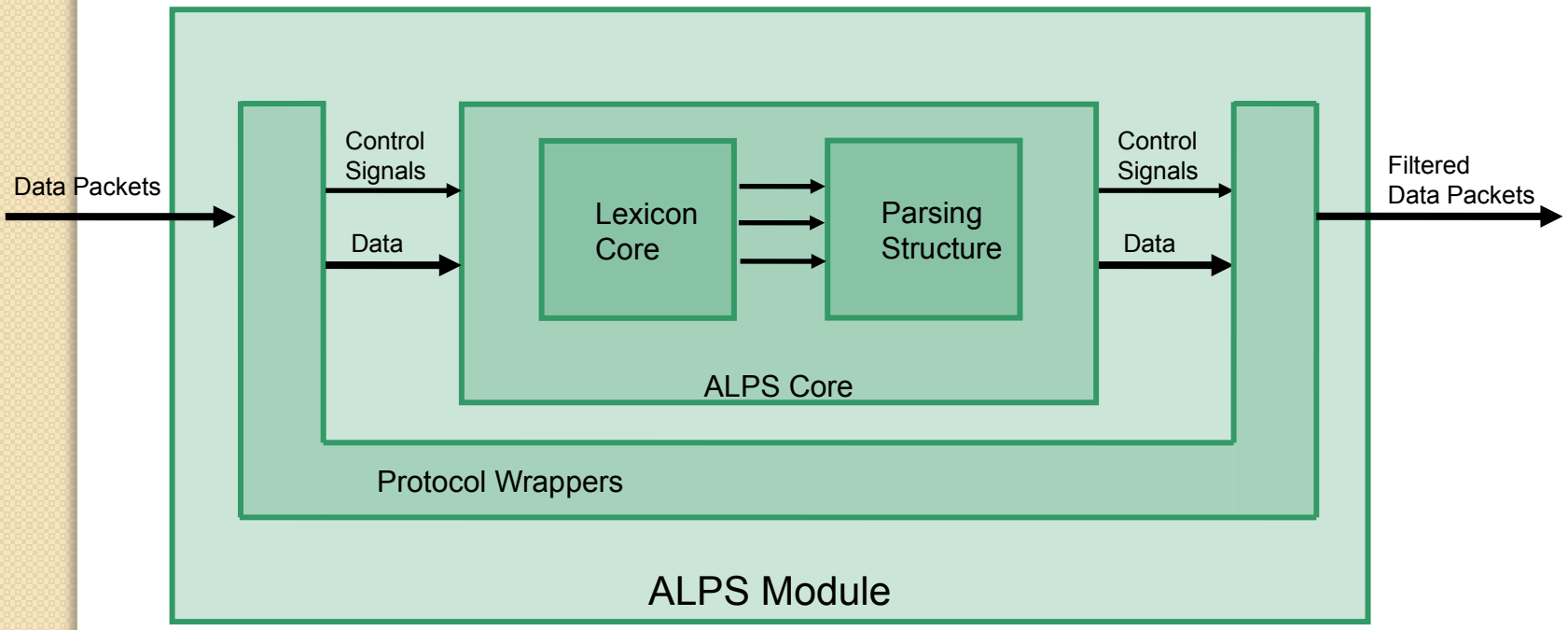


Parsing Structure

Terminal Symbols	*FOLLOW* Set
<cards>	<config>, <card>
</cards>	DONE
<config>	BITSTRING
BITSTRING	</config>
</config>	</cards>
<card>	<name>
</card>	<card>, </cards>
<name>	<first>
</name>	<title>
<first>	$STRING_{10}$
$STRING_{10}$	</first>
</first>	<last>
<last>	$STRING_{13}$
$STRING_{13}$	</last>
</last>	</name>
<title>	$STRING_{16}$
$STRING_{16}$	</title>
</title>	<phone>
<phone>	$STRING_{19}$
$STRING_{19}$	</phone>
</phone>	<desc>
<desc>	$STRING_{22}$
$STRING_{22}$	</desc>
</desc>	</card>



ALPS Module on FPX Platform



Example Data Packet

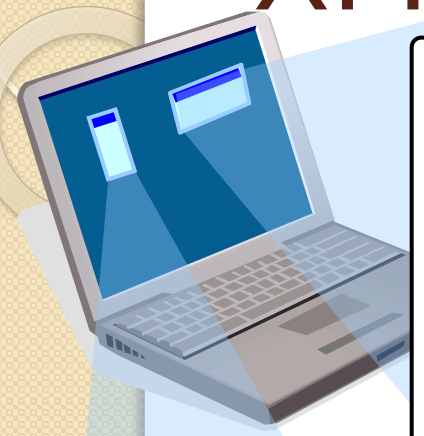
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE card SYSTEM "card.dtd">
<cards>0
 <card>5
  <name>7
   <first>9James10</first>11
   <last>12Moscola13</last>14
  </name>8
  <title>15student16</title>17
  <phone>18555-555-555519</phone>20
  <desc>21A graduate student at Washington University22</desc>23
 </card>6
</cards>1
```

- Red subscript indicates the token number in the hardware

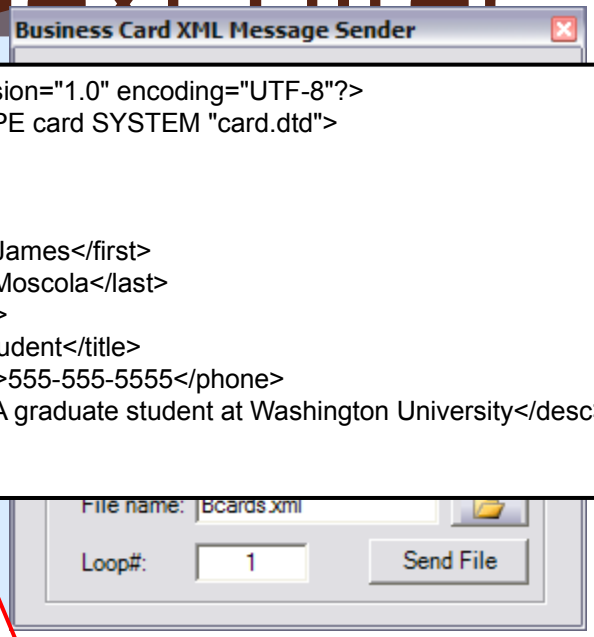
Data Packet Example

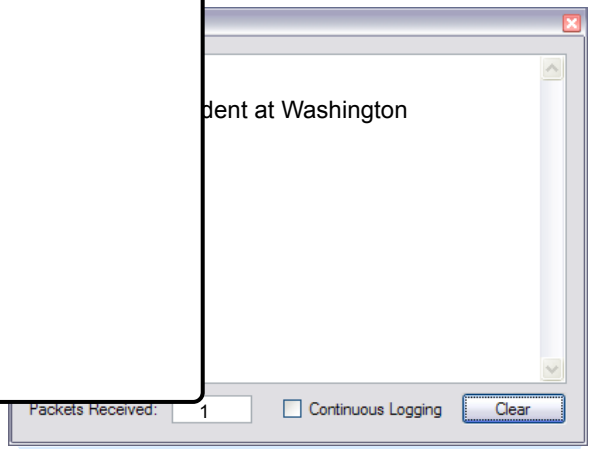
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE card SYSTEM "card.dtd">
<cards>
  <card>
    <name>
      <first>James</first>
      <last>Moscola</last>
    </name>
    <title>student</title>
    <phone>555-555-5555</phone>
    <desc>A graduate student at Washington University</desc>
  </card>
</cards>
```

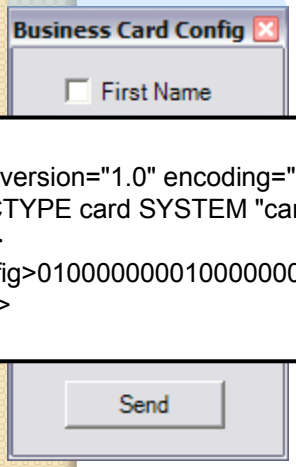
XML Text Filter



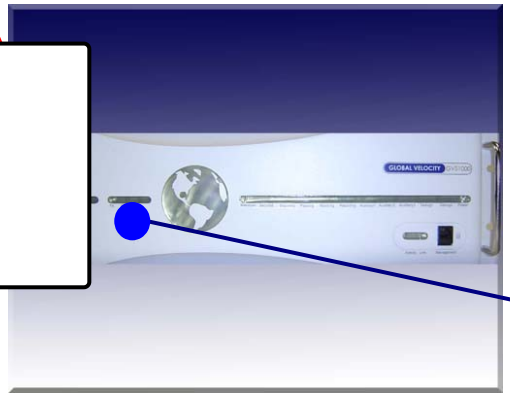
```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE card SYSTEM "card.dtd">
<cards>
 <card>
  <name>
   <first>James</first>
   <last>Moscola</last>
  </name>
  <title>student</title>
  <phone>555-555-5555</phone>
  <desc>A graduate student at Washington University</desc>
 </cards>
</card>
```








```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE card SYSTEM "card.dtd">
<card>
  <config>010000000010000000000000</config>
</card>
```



Configure Filter