Akera

the lightweight application server

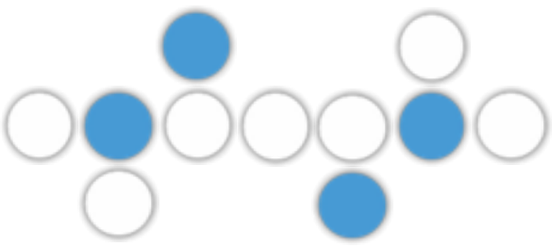
akera.io / acorn-it.com

Marian Edu

acornIT

what is it

- lightweight application server
- web enable your application
- node.js asynchronous I/O

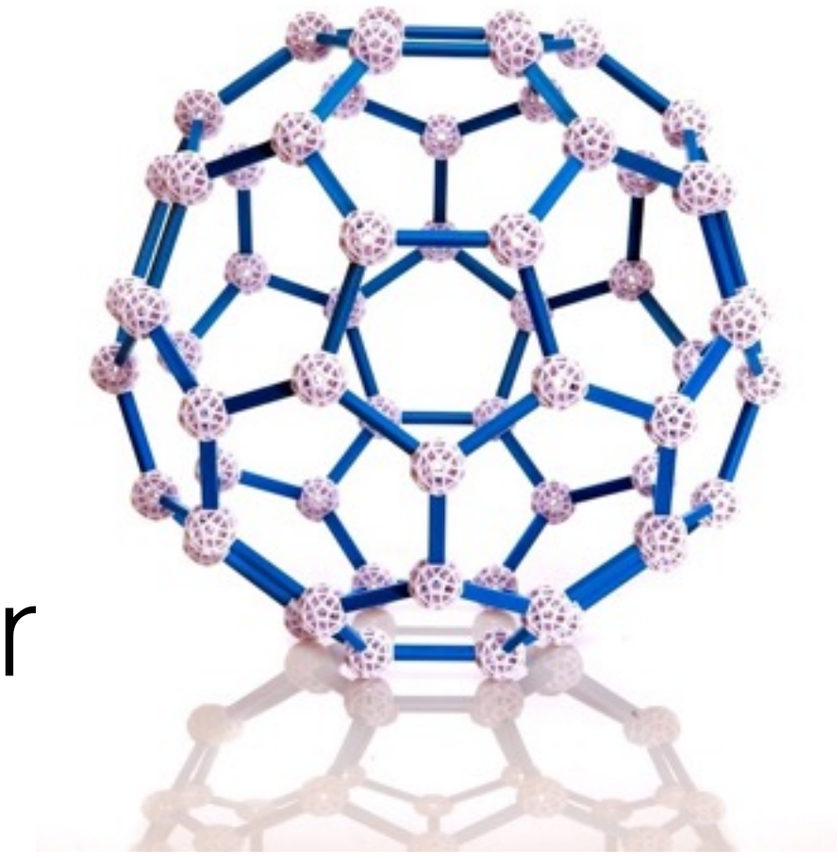


what it can do

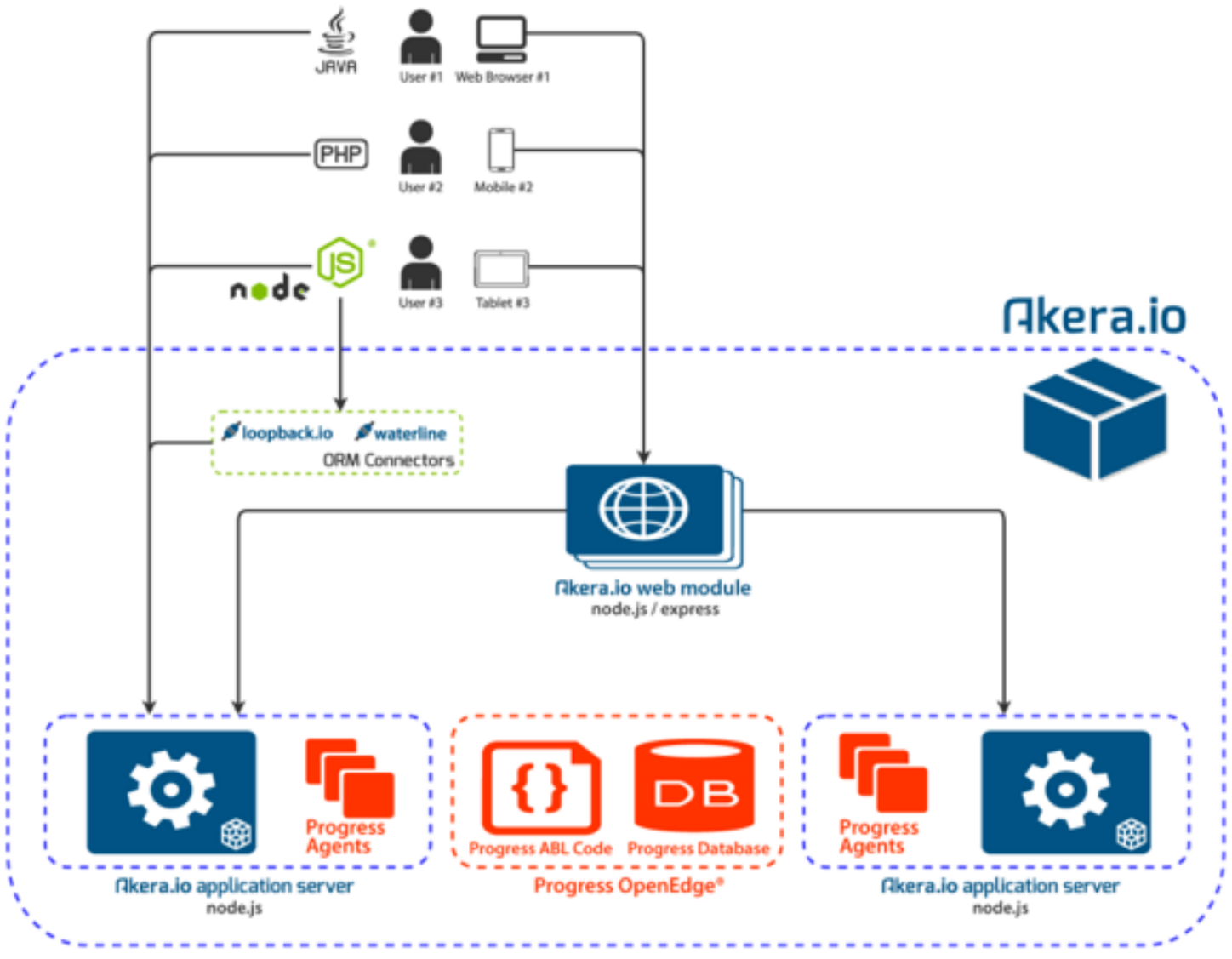
- webspeed support (speed-script)
- rest enablement (web / mobile)
- client API’s: node.js, java, php

architecture

- Integrate Functionality
- Distributed Architecture
- Modularity / Compositior



architecture



requirements

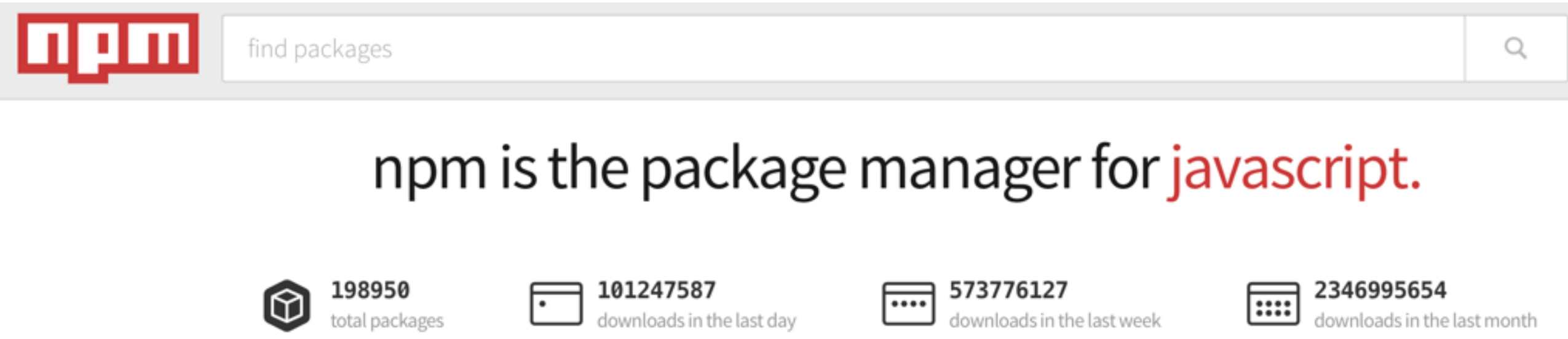
PROGRESS run-time / database

node.js

docker

why node.js

Node.js® is a JavaScript runtime built on Chrome's V8 JavaScript engine. Node.js uses an event-driven, non-blocking I/O model that makes it lightweight and efficient. Node.js' package ecosystem, npm, is the largest ecosystem of open source libraries in the world.



legacy

A. a computer system that has been in service for many years and that a business still relies upon, even though it is becoming expensive or difficult to maintain

B. left behind; old or no longer in active use

legacy

A. a computer system that has been in service for many years and that a business still relies upon, even though it is becoming expensive or difficult to maintain

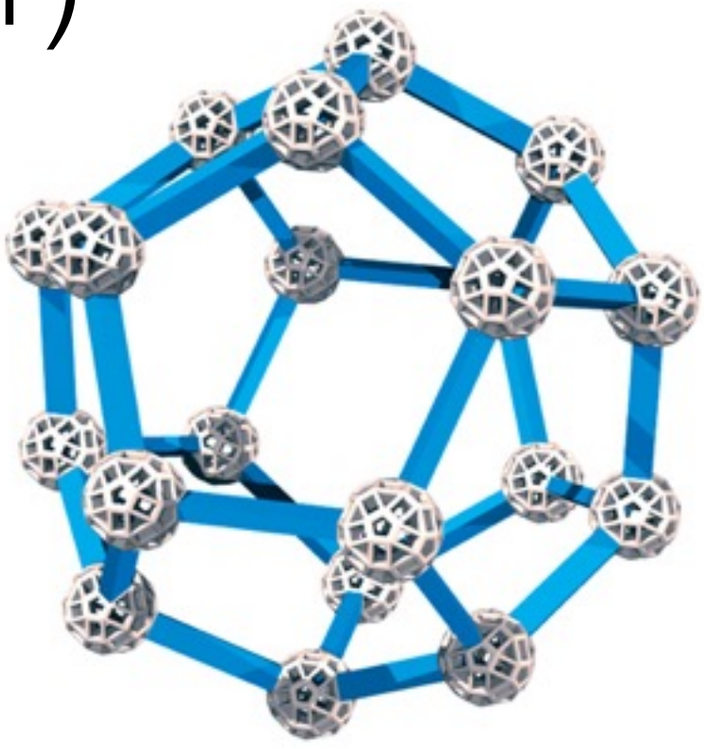
B. heritage

YOUR code

- procedural or object oriented
- UI separation required
- framework can help, not required
- Progress version 9.1A+

components

- broker (application server)
- web
- API
- ORM connectors



broker

- require Progress run-time/database
- one node.js process - broker
- several progress processes - agents
- multiple instances configurable
- global or user specific configuration

web

- node.js express middleware
- web interface for brokers
- load balancing between brokers (alias/weight)
- multiple instances configurable
- global or user specific configuration

web

- stand-alone or ‘mounted’ in any express app
- static files
- bootstrap initialisation scripts
- expose business logic using the API module
- development studio

api

- node.js package
- require(‘akera-api’);
- data access (CRUD)
- database meta-data
- business logic access (procedure/function)

loopback

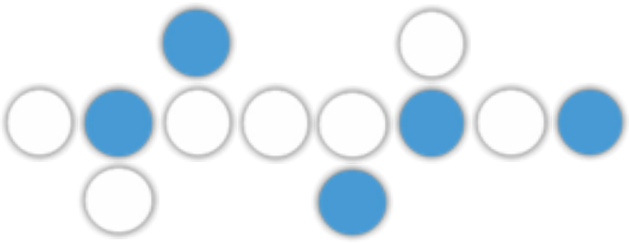
- ORM and more

StrongLoop
An IBM Company

- loopback connector
- data access (CRUD)
- database meta-data (discovery)

demo time

- akera-server
- akera-web
- akera-api
- akera-loopback-demo on npmjs
- akera-loopback-demo source code on github



online demo

http://akera.io/online-demo

broker

- npm set registry “http://repository.akera.io/“
- npm install -g akera-server
- akera-server -h /opt/akera config demo
- akera-server -h /opt/akera run demo

web

- npm set registry “http://repository.akera.io/“
- npm install -g akera-web
- akera-web -h /opt/akera config
- akera-web -h /opt/akera run
- http://localhost:8300/demo/studio

api

- npm set registry “http://repository.akera.io/“
- npm install akera-api

select

```
var akera = require('akera-api');
var f = akera.query.filter;

akera.connect('localhost', 8900).then(function(conn) {
   return conn;
}).then(
   function(conn) {
      conn.query.select(‘customer')
         .fields('custnum', 'name', ‘city', ‘address')
         .where(f.and(f.gt('custnum', 4), f.gt('balance', 5000)))
         .offset(5)
         .limit(10)
         .all().then(function(rows) {
            console.log(JSON.stringify(rows, null, '\t'));
         });
});
```

open query

```
var akera = require('akera-api');

akera.connect('localhost', 8900).then(function(conn)
{
   return conn;
}).then(
   function(conn) {
      conn.query.select(‘customer')
         .fields('custnum', 'name', ‘city', ‘address')
         .open().then(function(qry) {
            getNext(qry);
         });
});
```

find

```
var akera = require('akera-api');

akera.connect('localhost', 8900).then(function(conn)
{
   return conn;
}).then(
   function(conn) {
      conn.query.find(‘customer')
         .fields('custnum', 'name', ‘city', ‘address')
         .fetch().then(function(cust) {
         console.log(JSON.stringify(cust, null, '\t'));
      });
});
```

(up)insert

```
var akera = require('akera-api');

akera.connect('localhost', 8900).then(function(conn) {
    return conn;
}).then(
    function(conn) {
        conn.query.insert('customer')
            .set({
                custnum: 3030,
                name: 'Marian',
                city: 'Cluj',
                country: 'Romania',
                address: '48A Florilor'
                })
            .fetch().then(function(cust) {
                console.log(JSON.stringify(cust, null, '\t'));
            });
});
```

update

```
var akera = require('akera-api');
var f = akera.query.filter;

akera.connect('localhost', 8900).then(function(conn) {
    return conn;
}).then(
        function(conn) {
            conn.query.update(‘customer')
                .where(f.eq('custnum', 3035))
                .set( { ‘discount’, 20 })
                .go()
                .then(function(count) {
                        console.log('Updated ' + count + ' records.');
                        process.exit();
                    }, function(err) {
                        console.log('Error: ' + err.message);
                    });
    });
```

delete

```
var akera = require('akera-api');
var f = akera.query.filter;

akera.connect('localhost', 8900).then(function(conn) {
   return conn;
}).then(
      function(conn) {
         conn.query.destroy(‘customer')
            .where(f.eq('custnum', 3035))
            .go()
            .then(function(count) {
                  console.log('Deleted ' + count + ' records.');
                  process.exit();
               }, function(err) {
                  console.log('Error: ' + err.message);
               });
   });
```

run

```
var akera = require('akera-api');
var p = akera.call.parameter;

akera.connect('localhost', 8900).then(function(conn) {
    return conn;
}).then( function(conn) {
    conn.call.procedure('demo/sports/getCustBalance.p')
        .parameters(
                p.input(34),
                p.output('decimal'),
                p.output('decimal'),
                p.output('decimal'))
        .run().then(function(output) {
                    console.log(JSON.stringify(output, null, '\t'));
         }, function(err) {
            console.log(err);
         });
});
```

loopback

- npm set registry “http://repository.akera.io/“
- npm install loopback-connector-akera
- npm install loopback-discovery

```
"akera": {
    "host": "localhost",
    "port": 8900,
    "name": "akera",
    "useSSL": false,
    "connector": "akera",
    "database": "sports2000"
 }
```

loopback demo

- npm set registry “http://repository.akera.io/“
- npm install akera-loopback-demo
- cd node_modules/akera-loopback-demo
- slc run
- http://localhost:3333

what next

- REST Explorer
- Developer Studio (SPA)
- Tooling (Eclipse)
- JSDO integration
- Kendo Data Source for LoopBack

things to do

- Visit http://akera.io
- Mailing List
- Online Demo - http://akera.io/online-demo/
- Ask for a Demo/PoC - contact@akera.io

Thank You.