Beam me up, Scotty!

Rails for the "Enterprise"

Charles Johnson
Director, IS Applications
Centerstone

Rick Bradley
Project Manager
Centerstone

What Enterprise?

What Enterprise?

- Enterprise system?

What Enterprise?

- Enterprise system?
- Enterprise software?

What Enterprise?

- Enterprise system?
- Enterprise software?
- Enterprise platform?

Centerstone Enterprise

Centerstone Enterprise

- 25 Counties

Centerstone Enterprise

- 25 Counties
- 150 Locations

Centerstone Enterprise

- 25 Counties
- 150 Locations
- 40,000 Clients

Centerstone Enterprise

- 25 Counties
- 150 Locations
- 40,000 Clients
- 1,100 Staff

Centerstone Enterprise

- 25 Counties
- 150 Locations
- 40,000 Clients
- 1,100 Staff
- Comprehensive Electronic Record

Centerstone Enterprise

Requirements

Dimension

Centerstone Enterprise

Requirements

Dimension

Available & Reliable

Centerstone Enterprise

Requirements

Available & Reliable

Dimension

- All the time

Centerstone Enterprise

Requirements

Available & Reliable

Accessible

Dimension

- All the time

Centerstone Enterprise

Requirements	Dimension
Available & Reliable	• All the time
Accessible	• Everywhere

Centerstone Enterprise

Requirements	Dimension
Available & Reliable	• All the time
Accessible	• Everywhere
Scalable	

Centerstone Enterprise

Requirements	Dimension
Available & Reliable	• All the time
Accessible	• Everywhere
Scalable	• 300 to 9000 users

Centerstone Enterprise

Requirements

Dimension

Centerstone Enterprise

Requirements

Dimension

Maintenance

Centerstone Enterprise

Requirements

- Maintenance

Dimension

- Invisible

Centerstone Enterprise

Requirements

Maintenance

Dimension

- Invisible

Management

Centerstone Enterprise

Requirements

Maintenance

Dimension

- Invisible

Management

- Central

Centerstone Enterprise

Requirements

Maintenance

Management

Agnostic

Dimension

- Invisible
- Central

Centerstone Enterprise

Requirements

Maintenance

Management

Agnostic

Dimension

- Invisible
- Central
- Hardware

Centerstone Enterprise

Requirements	Dimension
Maintenance	• Invisible
Management	• Central
Agnostic	• Hardware • Database

“Enterprise”

“Big”

Bureaucratic

SOA

SOL

Vendor-driven definition -- the true source of the term "Enterprise".

Vendor-driven definition -- the true source of the term "Enterprise".

Java, XML, RDF

... EJB3.0, J2EE, JUnit

... Hibernate + XDoclet

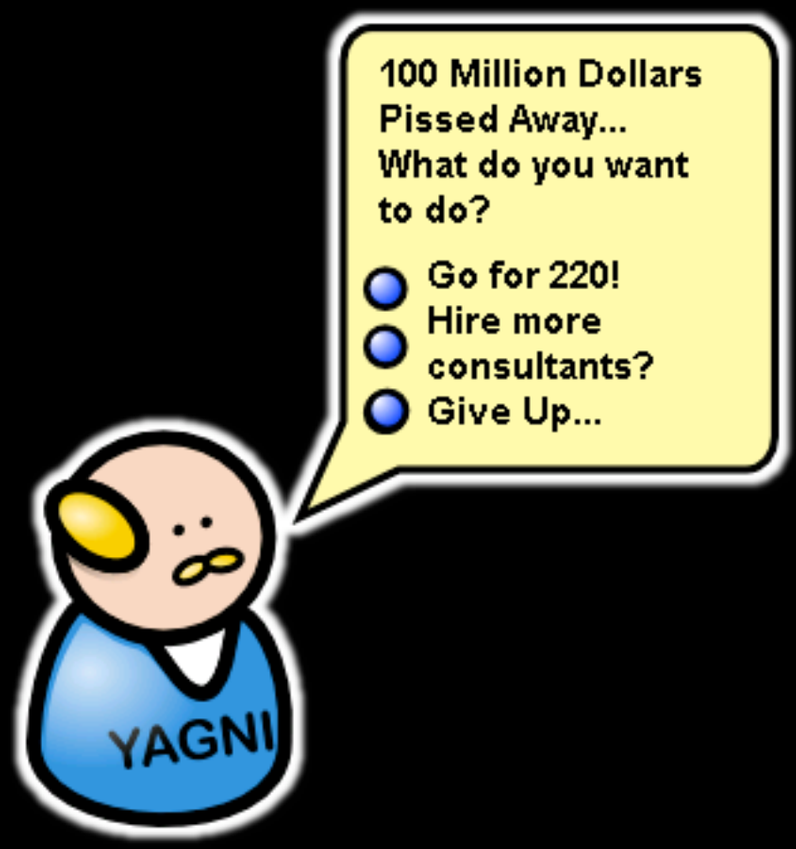
... n-phase commit

... RMI, XML-RPC, WSDL

... JMX, JAXL, SASL, CORBA

... WOTA, FRXN, JOKE

Aggressive!



-- with sincerest apologies to the creators of "Bunk and Rambling"...

Inverted

We don't use the Internet to track down a market for our product.

We have a fixed market for whom we build a new product.

Existing

Unlike “green field” applications, “Enterprise” applications fight the ills of the organization at least as much as technical hurdles.

Stubborn

Changing business processes (even horrid ones) can be very difficult.

@#$%!

Rick's First Rule of Enterpriseyness:

Any sufficiently large system left at rest in any sufficiently @#$%!-ed up environment will decay into a stack of electronic post-it notes.

Start

+

small team

- isolated offices

+

open source

open minds

-

Big Design Up Front

+

non-profit

-

for-profit

“collaborator”

+

web + DB

experience

-

Java + Oracle

+

support from the Penthouse

- widely distributed “customers”

+

camaraderie

- bad legacy habits...

-

no unit tests,
no functional tests,
no automated tests,
no automated build.

Goal

+

agility

small milestones, story-driven design, refactoring, pairing, few meetings, ...

+

domain

driven

design

in some ways we have to be careful not to end up BDUF. This can be meshed with "Getting Real" and TDD/BDD if done carefully. "Shared Domain Language" is critical. Avoid "too much talk, too little code."

+

solid testing

+

reproducibility

continous integration, one-word builds, automated deployments, automated upgrades and rollbacks, nightly data conversions, etc.

+

scalability

&

portability

"Share Nothing" horizontal scalability, lightweight software, fast commodity hardware; database agnosticism, browser agnosticism.

+++!!!

launch bonus:
a basket full of
psychotropics.

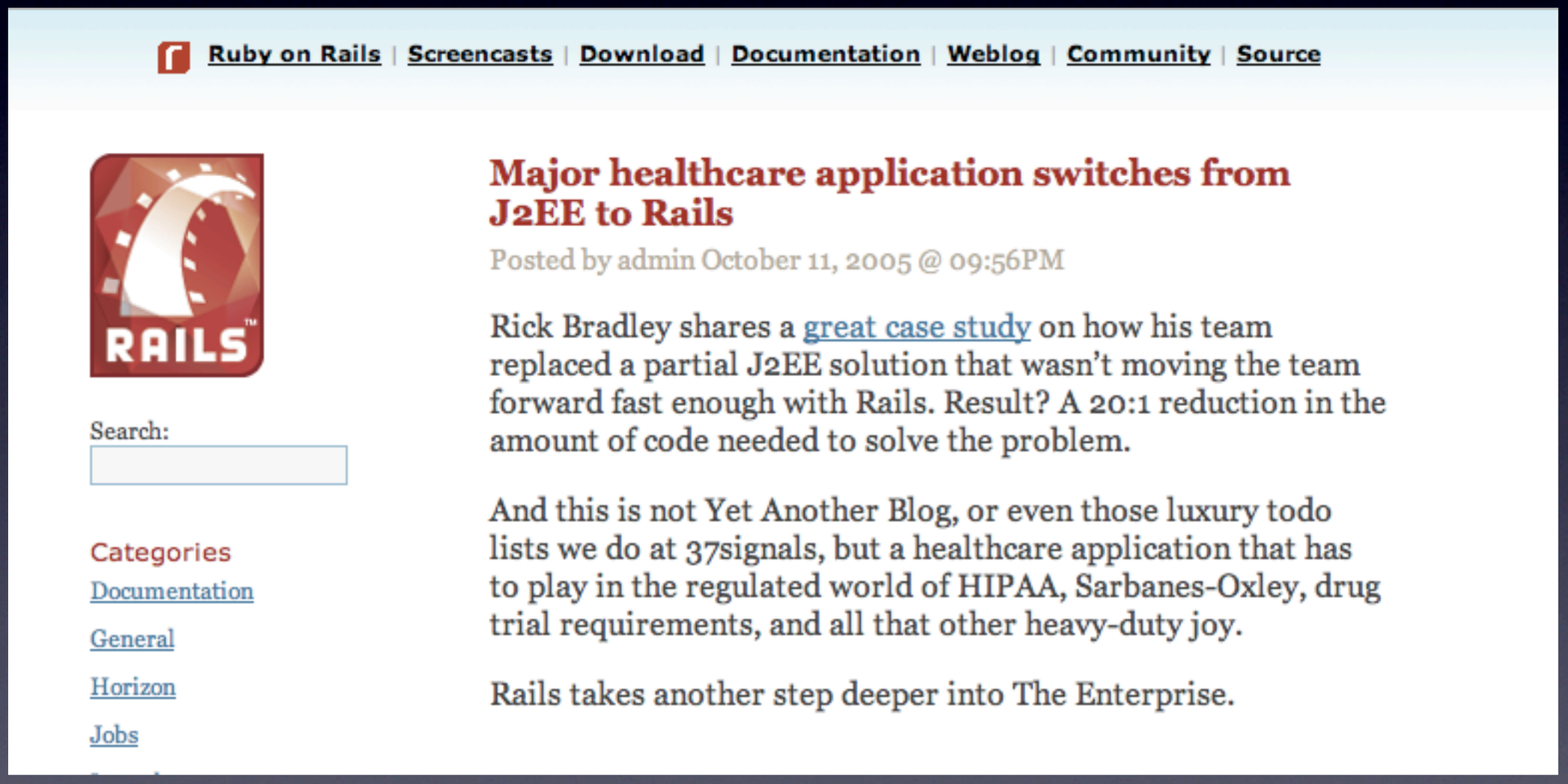
legacy system

250,000+ lines of embedded
“thedailywtf”-compliant
Oracle PLSQL code

java system

JBoss, Struts, JSP, Hibernate, EJB3 (draft),
JUnit, Ant, CruiseControl, ...

rails transition



http://weblog.rubyonrails.com/2005/10/11/major-healthcare-application-switches-from-j2ee-to-rails/

rails transition

Java:

10361 lines of Java code
1143 lines of JSP
8082 lines of XML
1267 lines of build configuration

20853 TOTAL lines of stuff

Rails:

494 lines of code (rake stats: 386 LOC)
254 lines of RHTML
75 lines of configuration
0 lines of build configuration
---
823 TOTAL lines of stuff

20+ : 1 code reduction
25+ : 1 "stuff" reduction

rails transition

8:1 reduction in books

rails today

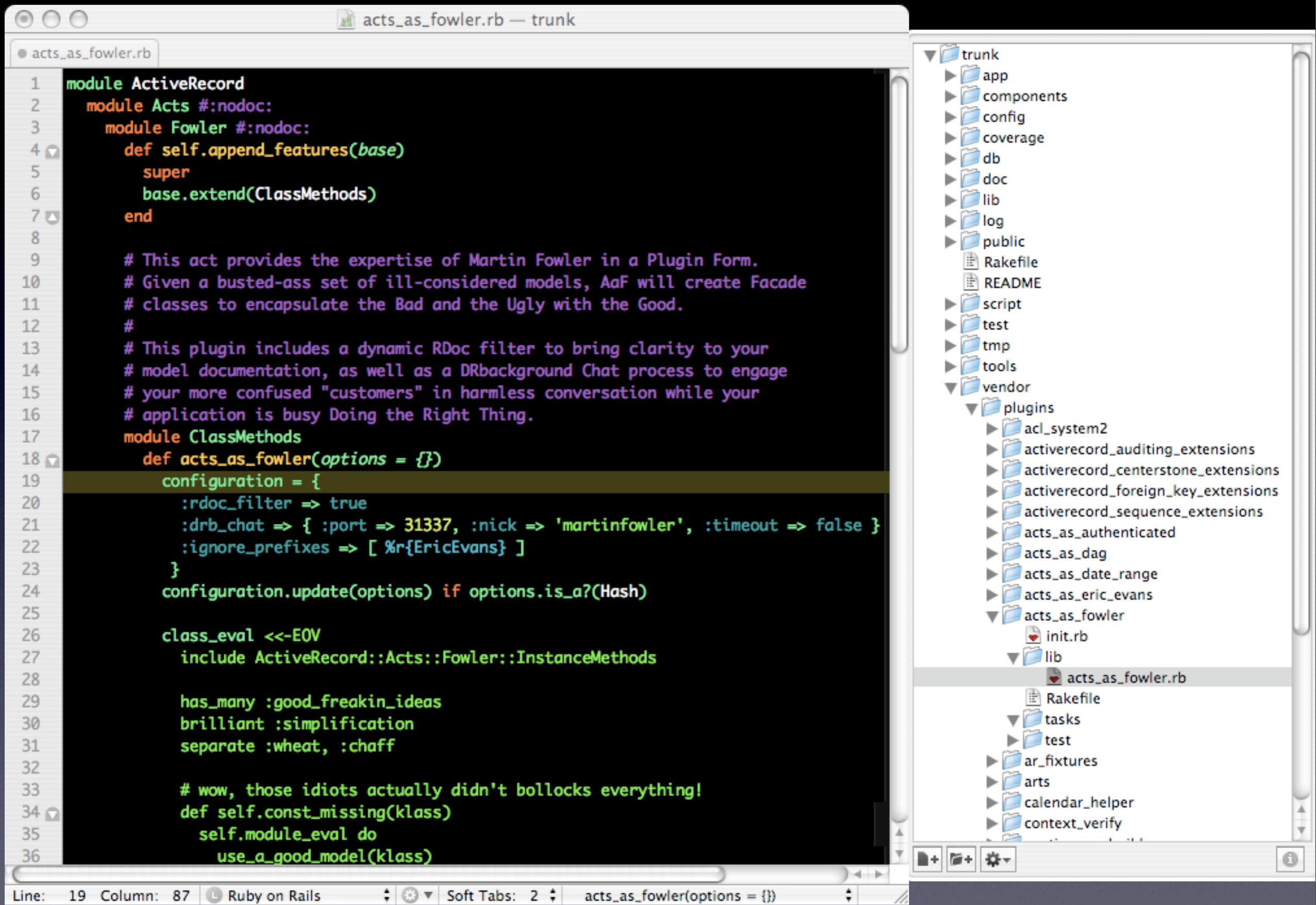
```
+----------------------+-------+-------+---------+---------+-----+-------+
| Name                 | Lines |   LOC | Classes | Methods | M/C | LOC/M |
+----------------------+-------+-------+---------+---------+-----+-------+
| Helpers              |   385 |   227 |       0 |      29 |   0 |     5 |
| Controllers          |   989 |   593 |      19 |      64 |   3 |     7 |
| Components           |     0 |     0 |       0 |       0 |   0 |     0 |
|   Functional tests   |  1110 |   815 |      31 |     117 |   3 |     4 |
| Models               |  1418 |   508 |      56 |      47 |   0 |     8 |
|   Unit tests         |  2218 |  1407 |      49 |     213 |   4 |     4 |
| Libraries            |     0 |     0 |       0 |       0 |   0 |     0 |
|   Integration tests  |    42 |    33 |       2 |       5 |   2 |     4 |
+----------------------+-------+-------+---------+---------+-----+-------+
| Total                |  6162 |  3583 |     157 |     475 |   3 |     5 |
+----------------------+-------+-------+---------+---------+-----+-------+
      Code LOC: 1328     Test LOC: 2255     Code to Test Ratio: 1:1.7
```

Techniques

domain
driven
design

analysis patterns

party, accountability, observation, phenomenon, measurement, plan, action, specification, time ranges, ...

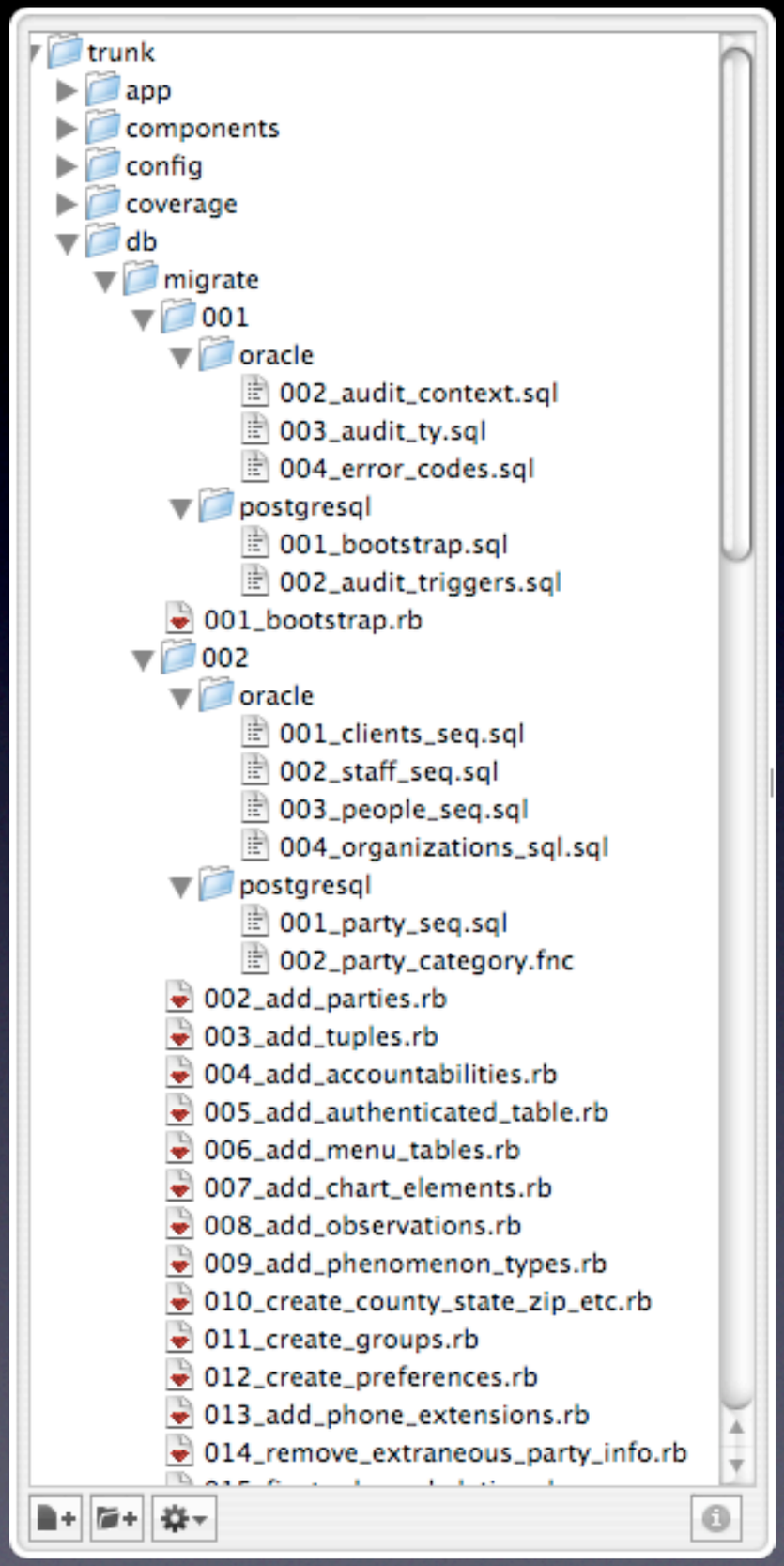


try to burn a little time here... talk about how finding “Analysis Patterns” was such a useful discovery, having gotten to meet Martin was a real boon to us, even exchanged some emails. Keeping Martin Fowler at the core of the system ended up being really beneficial to us.

database
“agnosticism”

...any database, as long as it’s Oracle or Postgres

to paraphrase Henry Ford: ....; Actually we may support SQLServer. We simply need to have an ACID db, with standard SQL support, with support for constraints and triggers sufficient to perform auditing.


trunk
app
components
config
coverage
db
migrate
001
oracle
002_audit_context.sql
003_audit_ty.sql
004_error_codes.sql
postgresql
001_bootstrap.sql
002_audit_triggers.sql
001_bootstrap.rb
002
oracle
001_clients_seq.sql
002_staff_seq.sql
003_people_seq.sql
004_organizations_sql.sql
postgresql
001_party_seq.sql
002_party_category.fnc
002_add_parties.rb
003_add_tuples.rb
004_add_accountabilities.rb
005_add_authenticated_table.rb
006_add_menu_tables.rb
007_add_chart_elements.rb
008_add_observations.rb
009_add_phenomenon_types.rb
010_create_county_state_zip_etc.rb
011_create_groups.rb
012_create_preferences.rb
013_add_phone_extensions.rb
014_remove_extraneous_party_info.rb

001_bootstrap.sql — trunk

001_bootstrap.sql

```
1  CREATE FUNCTION plpgsql_call_handler() RETURNS language_handler AS
2      '$libdir/plpgsql' LANGUAGE C;
3
4  CREATE FUNCTION plpgsql_validator(oid) RETURNS void AS
5      '$libdir/plpgsql' LANGUAGE C;
6
7  CREATE TRUSTED PROCEDURAL LANGUAGE plpgsql
8      HANDLER plpgsql_call_handler
9      VALIDATOR plpgsql_validator;
10
11 CREATE OR REPLACE FUNCTION get_auth_party() returns integer
12   AS '/usr/local/lib/audit', 'get_auth_party'
13   LANGUAGE C STRICT;
14
15 CREATE OR REPLACE FUNCTION set_auth_party(integer) returns integer
16   AS '/usr/local/lib/audit', 'set_auth_party'
17   LANGUAGE C STRICT;
18
19 create or replace function set_audit_context(p_userid varchar) returns varchar
20   as $set_audit_context$
21
22 begin
23   perform set_auth_party(p_userid);
24   return '';
25 end;
26 $set_audit_context$ language plpgsql;
27
28 create or replace function audit_info() returns varchar as
29 $audit_info$
30 declare
31   result varchar(200);
32 begin
33   result :=
34     '--- '||chr(10)||
35     'os_timestamp: '||to_char(localtimestamp, 'YYYY-MM-DD HH24:MI')||chr(10)||
36     'party_id: '    ||get_auth_party()                              ||chr(10)||
```

Line: 1 Column: 1 SQL Tab Size: 4 plpgsql_call_handler

application.rb — trunk

001_bootstrap.sql | auditing_extensions.rb | application.rb | authenticated_system.rb

```ruby
#
#--
#    Copyright (c) 2005 Centerstone, All rights reserved.
#
#    This program is free software; you can redistribute it and/or modify
#    it under the terms of the Centerstone Public License as published by
#    the Centerstone; either version 1 of the License, or
#    (at your option) any later version.
#
#    This program is distributed in the hope that it will be useful,
#    but WITHOUT ANY WARRANTY; without even the implied warranty of
#    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#    Centerstone Public License for more details.
#
#    You should have received a copy of the Centerstone Public License
#    along with this program; if not, write to Centerstone, P.O. Box 40406
#    Nashville, TN 37204-0406
#++
# Filters added to this controller will be run for all controllers in the application.
# Likewise, all the methods added will be available for all controllers.
class ApplicationController < ActionController::Base
  include AuthenticatedSystem

  before_filter :login_required,  :except => [:login]
  before_filter :set_defaults, :except => [:login]

  def rescue_action_in_public(exception) ...

protected
  def set_defaults
    Party.login_user_id = self.current_login.party.id
    ActiveRecord::Base.connection.audit_context = self.current_authenticated_login.party.id
    session[:context] ||= HashWithIndifferentAccess.new
  end
end
```

Line: 41 Column: 14 | Ruby on Rails | Soft Tabs: 2 | set_defaults

searching

staff.rb — trunk

staff.rb | application_helper.rb | search_controller.rb

```
21  # The Staff model is a Party that represents an employee of the organization deploying the system.
22  class Staff < Party
23    validates_presence_of         :surname
24    validates_uniqueness_of       :social_security_number
25    validates_format_of           :social_security_number, :with => /\d{9}/
26    validates_in_tuple            'core_services/staff/eeoc_race_sex', :eeoc_race_sex
27    validates_in_tuple            'core_services/staff/eeoc_job_category', :eeoc_job_category
28
29    searchable :internal_id, :name, :middle_name, :surname, :suffix, :birthdate, :social_security_number
30  end
```

Line: 29 Column: 1 | Ruby on Rails | Soft Tabs: 2 | Staff < Party

searchable_models plugin provides the ‘searchable’ DSL as well as a search controller, and a helper to let you put simple searches anywhere for any class.

Graphs

phenomenon_type.rb — trunk

staff.rb | phenomenon_type.rb | application_helper.rb | search_controller.rb

```
20  #
21  # A PhenomenonType is a concept about which ChartNote, Measurement,
22  # and CategoryObservations can be recorded with a ChartElement for a Client.
23  # The PhenomenonType collection forms a Directed Acyclic Graph (DAG) of subtypes
24  # and supertypes.  An observation of the presence of a subtype implies the presence
25  # of the supertypes.  An observation of the absence of a supertype implies the
26  # absence of the subtypes.
27  class PhenomenonType < ActiveRecord::Base
28    acts_as_dag :in => :subtypes, :out => :supertypes
29    validates_uniqueness_of :title
30    validates_presence_of   :title
31  end
```

Line: 28 Column: 48 | Ruby on Rails | Soft Tabs: 2 | PhenomenonType < ActiveRecord::Base

Graphs

acts_as_dag.rb — trunk

acts_as_dag.rb

```ruby
module ActiveRecord
  module Acts #:nodoc:
    module DAG #:nodoc:
      def self.append_features(base)
        super
        base.extend(ClassMethods)
      end

      # This act provides the capabilities for a class to behave as a directed acyclic graph.
      # The class that has this specified needs to have a <model>_edges tables defined.
      module ClassMethods
        def acts_as_dag(options = {})
          configuration = {
            :edges_table => "#{self.table_name}_edges",
            :in => :in,
            :out => :out,
          }
          configuration.update(options) if options.is_a?(Hash)

          class_eval <<-EOV
            include ActiveRecord::Acts::DAG::InstanceMethods

            has_and_belongs_to_many :#{configuration[:in]},
              :join_table => '#{configuration[:edges_table]}',
              :foreign_key => 'destination_id',
              :association_foreign_key => 'source_id',
              :class_name => '#{self.name}'

            has_and_belongs_to_many :#{configuration[:out]},
              :join_table => '#{configuration[:edges_table]}',
              :foreign_key => 'source_id',
              :association_foreign_key => 'destination_id',
              :class_name => '#{self.name}'
```

Line: 69 Column: 45 Ruby on Rails Soft Tabs: 2 acts_as_dag(options = {})

stories

Stories...

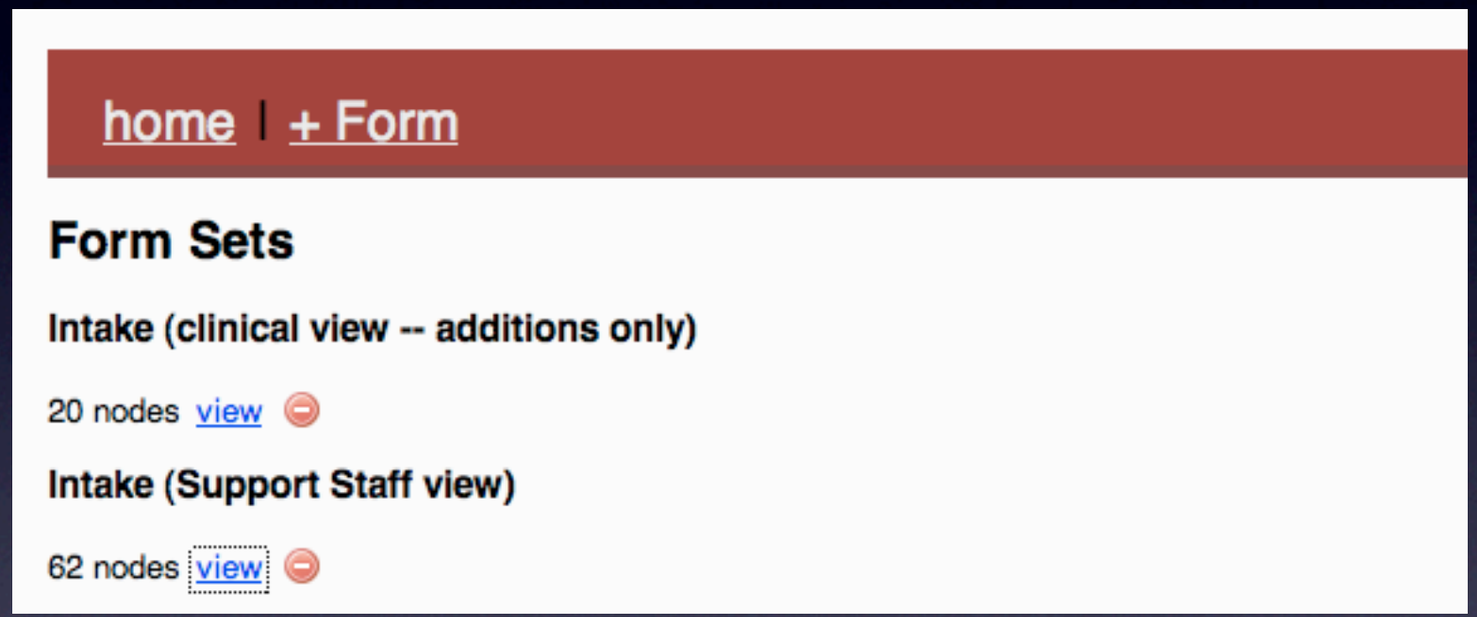
Stories | Tags | Login

Pages: 1 2 3

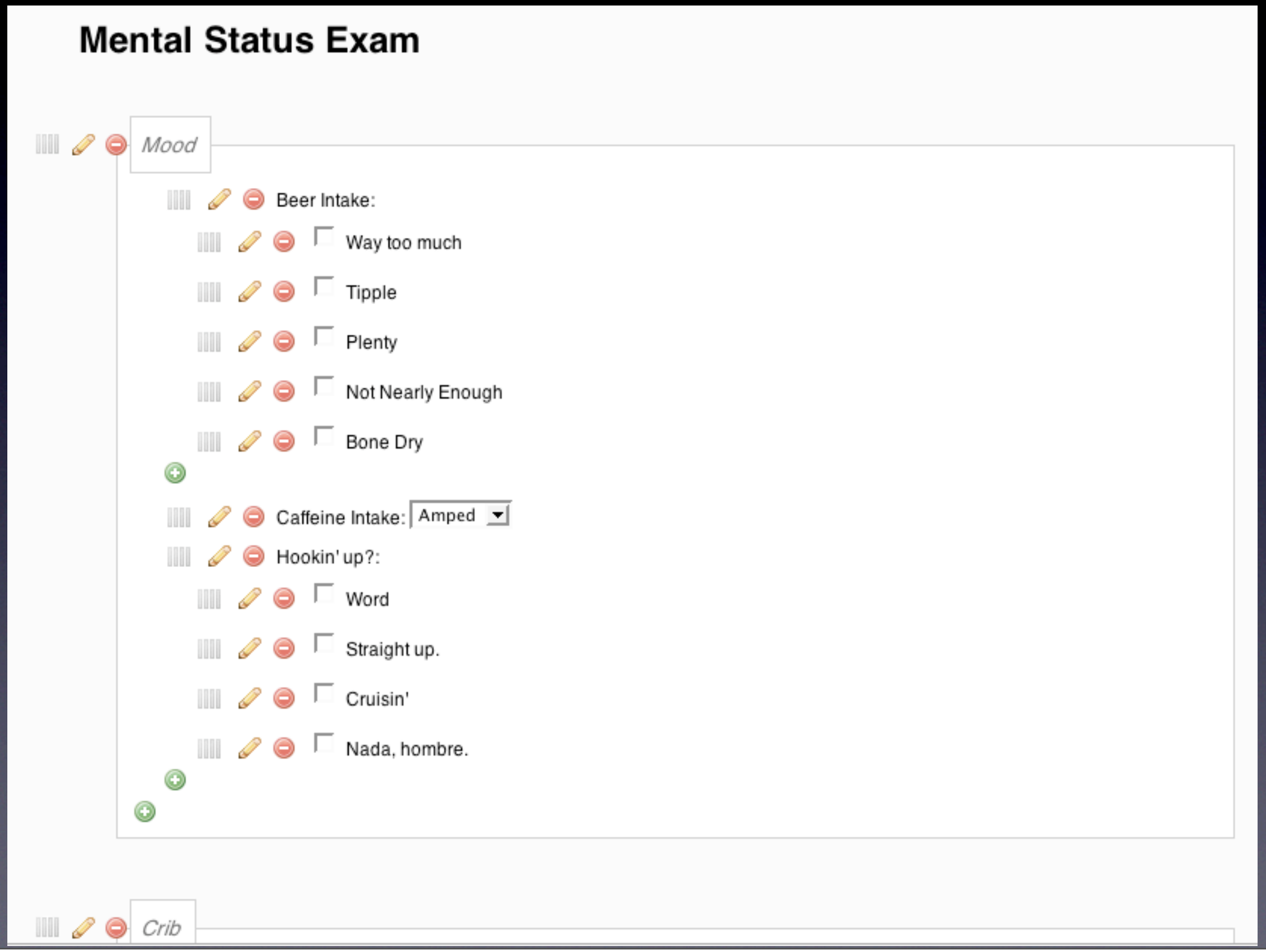
- Using Stories
- Client Summary: Current Meds
- Client Summary: Dosage for Meds
- Client Summary: Name for Meds
- Client Summary: Sources for Meds
- Client Summary Diagnosis
- Client Summary: Client Demographics
- Client Summary: Progress Note Listing
- Client Summary: Medication Allergy List
- Client Summary: Lab List
- Client Summary: Observation Graph
- VNS Note
- Client Creation should warn when duplicate info appears
- Protocol suggests observations to monitor
- Med Note: Client Demographics
- Med Note: Identifying Data
- Med Note: Chief Complaint
- Alter diagnosis from note (or eval)
- Med Note: Chief Complaint (Subjective)
- Alter Medications via the Progress Note

Form mocking

Form mocking



Form mocking



ActiveRecord magic

auditing

Level 0 - No auditing.

Level 1 - Read-only. Audit creation.

Level 2 - No deletion. Audit creation, changes.

Level 3 - Archive all changes and deletions.

018_basic_billing_and_credentialing.rb — trunk

008_add_observations.rb | 018_basic_billing_and_credentialing.rb

```
93
94    # A Credential is the authorization by a Panel for a Staff to deliver services
95    # at a given Location for a period of time.  It is contingent on the Staff mainto
96    # a certain licensure.
97    create_table :credentials, :audit => 2, :options => 'tablespace billdata' do |t|
98      t.column :licensure_id, :integer
99      t.column :staff_id, :integer
100     t.column :panel_id, :integer
101     t.column :location_id,  :integer
102     t.column :begin_date, :datetime
103     t.column :end_date, :datetime
104     t.column :created, :string, :limit => 200
105     t.column :updated, :string, :limit => 200
106   end
107
108   # A Rate is the price a payer will pay for a given service (identified by service
109   # to be delivered.
110   create_table :rates, :audit => 2, :options => 'tablespace billdata' do |t|
111     t.column :service_code_id, :integer
112     t.column :payer_id, :integer
113     t.column :begin_date, :datetime
114     t.column :end_date, :datetime
115     t.column :created, :string, :limit => 200
116     t.column :updated, :string, :limit => 200
117   end
118
119   # Actual delivered services.  An Activity performed for a Client, by a Staff, at
120   # Location, with a time reference.
121   create_table :services, :audit => 2, :options => 'tablespace billdata' do |t|
122     t.column :activity_id, :integer
123     t.column :location_id, :integer
124     t.column :staff_id, :integer
125     t.column :client_id, :integer
126     t.column :begin_date, :datetime
127     t.column :end_date, :datetime
```

Line: 50 Column: 1 | Ruby on Rails | Soft Tabs: 2 | self.up

foreign key extensions

acts_as_date_range

USPS

county, state, ZIP models
CD-ROM import scripts

Pitfalls

N.I.H.

syndrome

Refusing to use good code simply because it wasn't written by us. With Ruby sometimes it / is/ faster to write it ones self than use a plugin. Fortunately, as the plugin space has matured this has become less of an issue for us.

I. H.

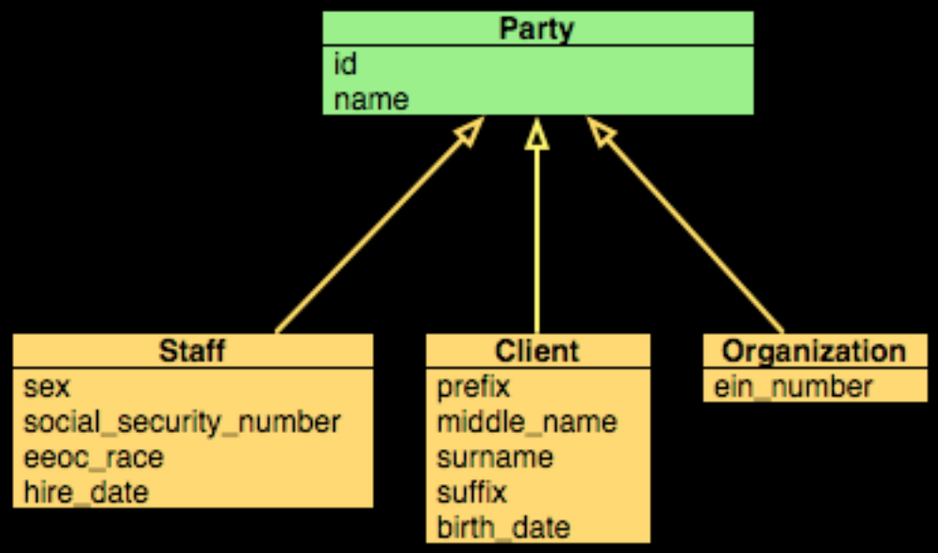
syndrome

feeling compelled to write a bit of functionality. when someone else, two weeks later, releases a plugin for the same thing, feeling compelled to adopt it. Also, overturning homegrown methodologies to do things The Rails Way. Contrast: moving from homegrown Login to Acts_as_authenticated-generated homegrown login with moving from SQL table scripts to Migrations.

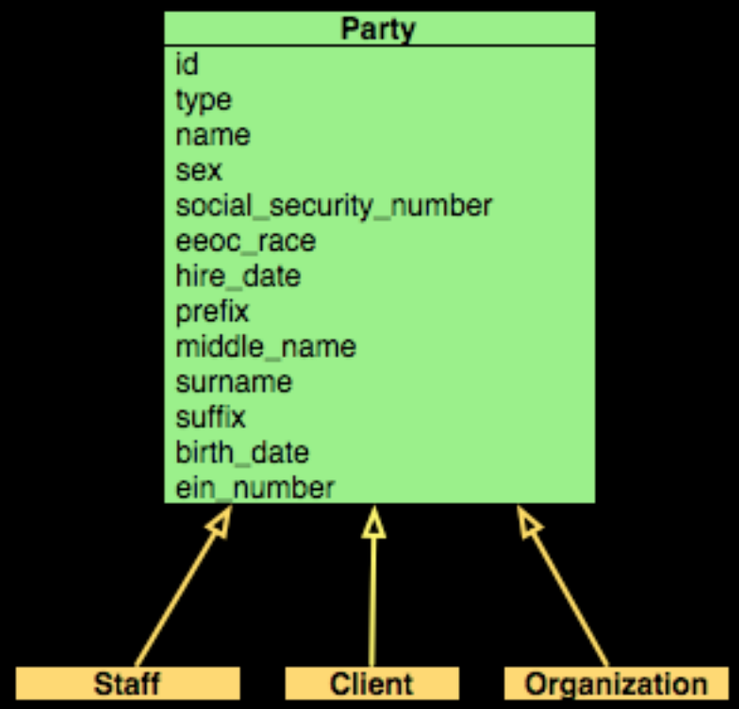
CTI

In implementing Fowler's "Party" and "Accountability" patterns, we attempted to use Class Table Inheritance with Rails. Getting inherited behavior, much less cross-database, was a nightmare.

parties in class table inheritance



parties in single table inheritance



Downsides: we still have report writers, outside processes working against database. Makes it difficult to allow access to some identifying data (for example, staff data) without giving access to client data. We were planning on using exports to reporting database anyway, but this requires us to think more carefully about views, etc., to control reporting access.

Attempts at CTI failed -- while possible to get the database semantics correct, synchronizing the separate tables (esp. efficiently, was a nightmare -- esp.

Bleeding Edge

In a number of areas we were able to identify just whose blood was on the bleeding edge.

-

Oracle

performance issues, primary key names, performance, adding/removing constraints, testing with constraints, synonyms, Windows-only/Oracle-only date problem, etc.

Nearly all the oracle people you find want to show you how cool they are by writing a bunch of non-portable Oracle crap, half of which doesn't work half as well as half of them think. As for standards-compliance and ease of use, Oracle blows.

Ticket #2791 (defect)

Windows-only Oracle-only pre-epoch date problem

Opened 7 months ago

Status: closed (fixed)

Last modified 3 months ago

Reported by:	rick@rickbradley.com	Assigned to:	Michael Schoen <schoenm@earthlink.net>
Priority:	normal	Milestone:	
Component:	ActiveRecord	Version:	0.14.3
Severity:	normal	Keywords:	oracle oci windows date epoch 1970
Cc:			

We had a lot of trouble even believing this problem existed, but we finally narrowed it down to a reproducible test. This problem does not exhibit itself under Linux, but does show up under Windows.

When running Rails 0.14.3 (and 0.14.2, and possibly 0.14.1), under Ruby 1.8.3 (and also under Ruby 1.8.2), against Oracle (10g, but I don't think the problem is with the Oracle version), on Windows XP (maybe more versions) we see an ArgumentError? exception ("time out of range") when trying to assign string date values to an AR date field prior to 1/1/1970 (the Unix epoch). This does NOT happen on Linux when all code and software versions are otherwise identical. We have tested this on 2 separate Windows systems (not ghosted copies -- long-running developer systems with different install and configuration histories) and multiple Linux systems of varying distributions.

The details:

Here is the Oracle table definition for our Party class:

```
create table party
(
  id                    number(12)           not null,
  lock_version          number(12) default 1 not null,
  category              varchar(30)          not null,
  name                  varchar(30)          not null,
  middle_name           varchar(30)
```


here's a fun one: oracle 10g, with rails running on Windows, can't store dates prior to the UNIX(!) epoch.

+

monkey-patching

active_record_ext.rb — phoenix

active_record_ext.rb

```
87  ##############################################################################
88  #
89  #  "PATCHES"
90  #
91  ##############################################################################
92
93  if $".grep(%r,oci8.rb,).length > 0
94    STDERR << "Patching Oracle columns() performance (see: http://dev.rubyonrails.org/ticket/3210)\n"
95    require 'oci8'
96
97    module ActiveRecord
98      module ConnectionAdapters
99        class OCIAdapter
100         def columns(table_name, name = nil) #:nodoc:
101           table_info = @connection.describe_any(table_name)
102
103           table_cols = %Q{
104             select column_name, data_type, data_default, nullable,
105                    decode(data_type, 'NUMBER', data_precision,
106                                      'VARCHAR2', data_length,
107                                      null) as length,
108                    decode(data_type, 'NUMBER', data_scale, null) as scale
109               from all_tab_columns
110              where owner      = '#{table_info.schema}'
111              and   table_name = '#{table_info.name}'
112           }
113           select_all(table_cols, name).map do |row|
114             row['data_default'].sub!(/^'(.*)'\s*$/, '\1') if row['data_default']
115             OCIColumn.new(
116               oci_downcase(row['column_name']),
117               row['data_default'],
118               row['data_type'],
119               row['length'],
120               row['scale'],
121               row['nullable'] == 'Y'
122             )
```

Line: 1 Column: 1 Ruby on Rails Soft Tabs: 2

+

great upstream

when we wrote a ticket, sent in a patch, we dealt directly (and immediately) with the maintainers of those functional areas. When the patch landed and we pulled edge or a new release, we just removed our monkey-patch.

+

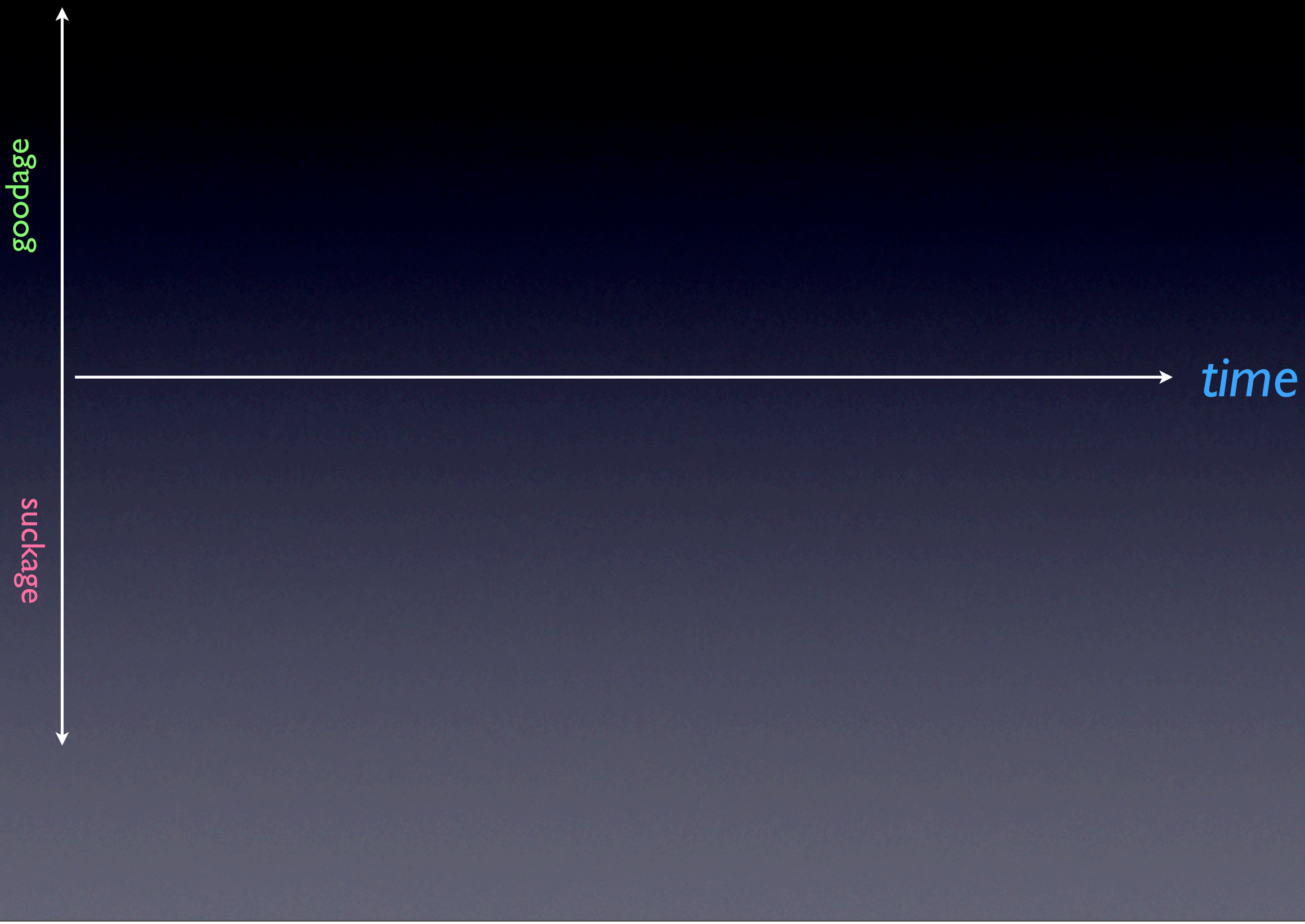
plugins

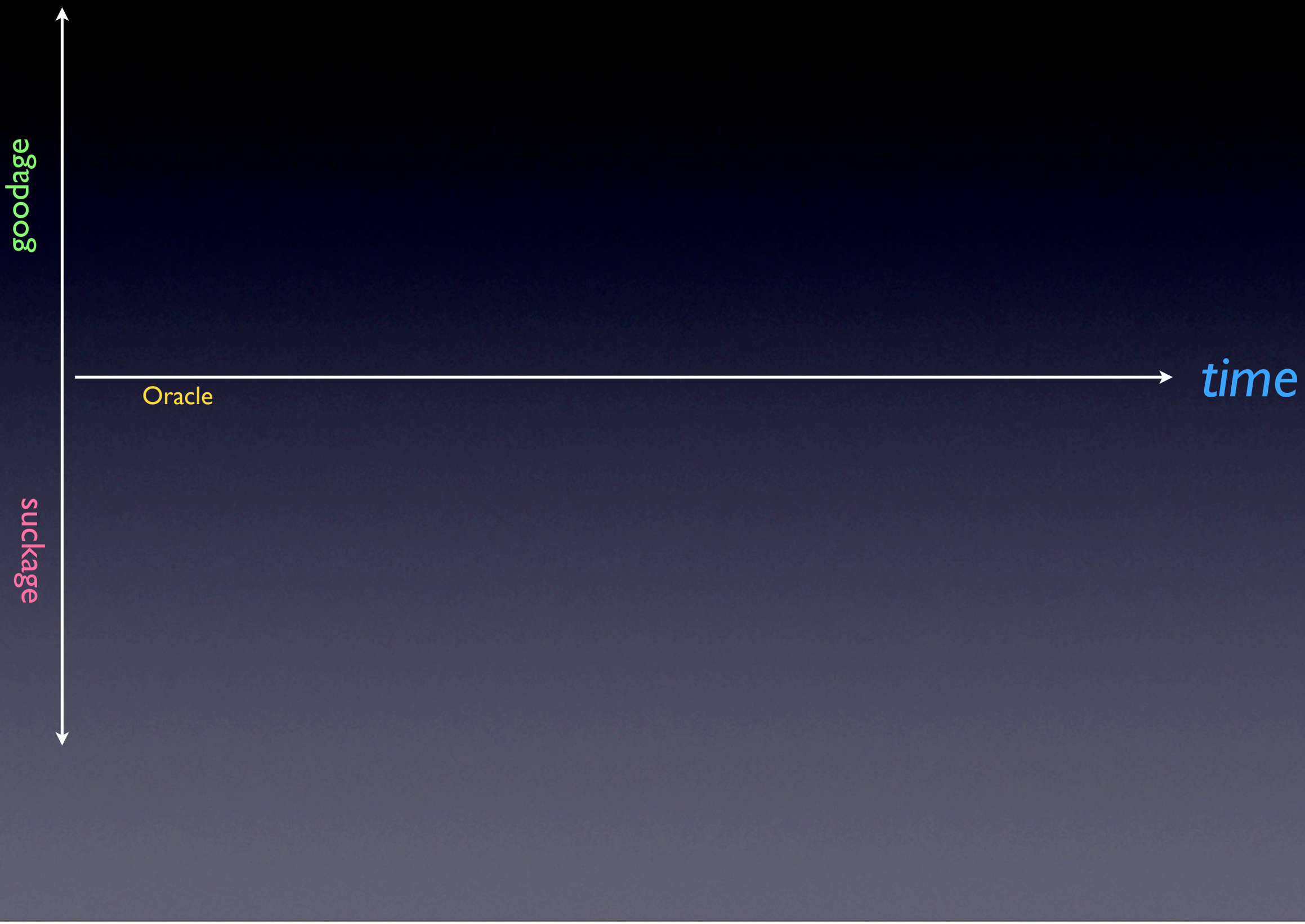
Now, instead of monkey-patching, we refactor fixes to plugins, which are a clean point of change and which can be easily removed when patches land in trunk.

Maximum Rate of Change

Less than the maximal rate of change? The obvious risk would be that the change by the target date would be less than ideal. The real risk is that without an aggressive change rate, NO real change would take place.

Faster than maximal? You risk confusing the team members, failing to provide enough stability to allow progress. Worse, you may awaken politically entrenched opponents who will cause harm to the project.


goodage
suckage
time


goodage
suckage
Oracle
time

goodage

suckage

Oracle

time

Big Design Up Front

goodage

suckage

Oracle

time

Big Design Up Front

“deployments”

goodage

suckage

Oracle

time

Big Design Up Front

"deployments"

for-profit partner

goodage

suckage

Oracle

time

Big Design Up Front

"deployments"
Java

for-profit partner

goodage

suckage

cvs

Oracle

time

Big Design Up Front

“deployments”
Java

for-profit partner

goodage

suckage

cvs

Oracle

time

SQL schemae

Big Design Up Front

“deployments”
Java

for-profit partner

goodage

suckage

cvs

Oracle

time

SQL schemae

Big Design Up Front

JBoss

“deployments”

Java

for-profit partner

goodage

suckage

Postgres

cvs

Oracle

time

SQL schemae

Big Design Up Front

JBoss

“deployments”

Java

for-profit partner

goodage

suckage

Postgres

cvs

Oracle

time

SQL schemae

Big Design Up Front

JBoss

“deployments”

Java

Hibernate

for-profit partner

goodage

suckage

time

svn

Postgres

Oracle

SQL schemae

Big Design Up Front

JBoss

“deployments”

Java

Hibernate

for-profit partner

goodage

suckage

svn

Postgres

Oracle

time

SQL schemae

Big Design Up Front

JBoss

Ant

“deployments”

Java

Hibernate

for-profit partner

goodage

suckage

svn

Postgres

Oracle

time

SQL schemae

Big Design Up Front

JBoss

Ant

“deployments”

Java EJB3.0

Hibernate

for-profit partner

goodage

suckage

time

svn

trac

Postgres

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JBoss

Ant

“deployments”

Java

EJB3.0

Hibernate

for-profit partner

goodage

suckage

time

analysis patterns

svn

trac

domain driven design

Postgres

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JBoss

Ant

“deployments”

Java

EJB3.0

Hibernate

for-profit partner

goodage

suckage

time

analysis patterns

continuous integration

svn

trac

domain driven design

Postgres

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JBoss

CruiseControl

Ant

“deployments”

Java

EJB3.0

Hibernate

for-profit partner

goodage

suckage

time

analysis patterns

continuous integration

svn

trac

domain driven design

Postgres

IRC

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JBoss

CruiseControl

Ant

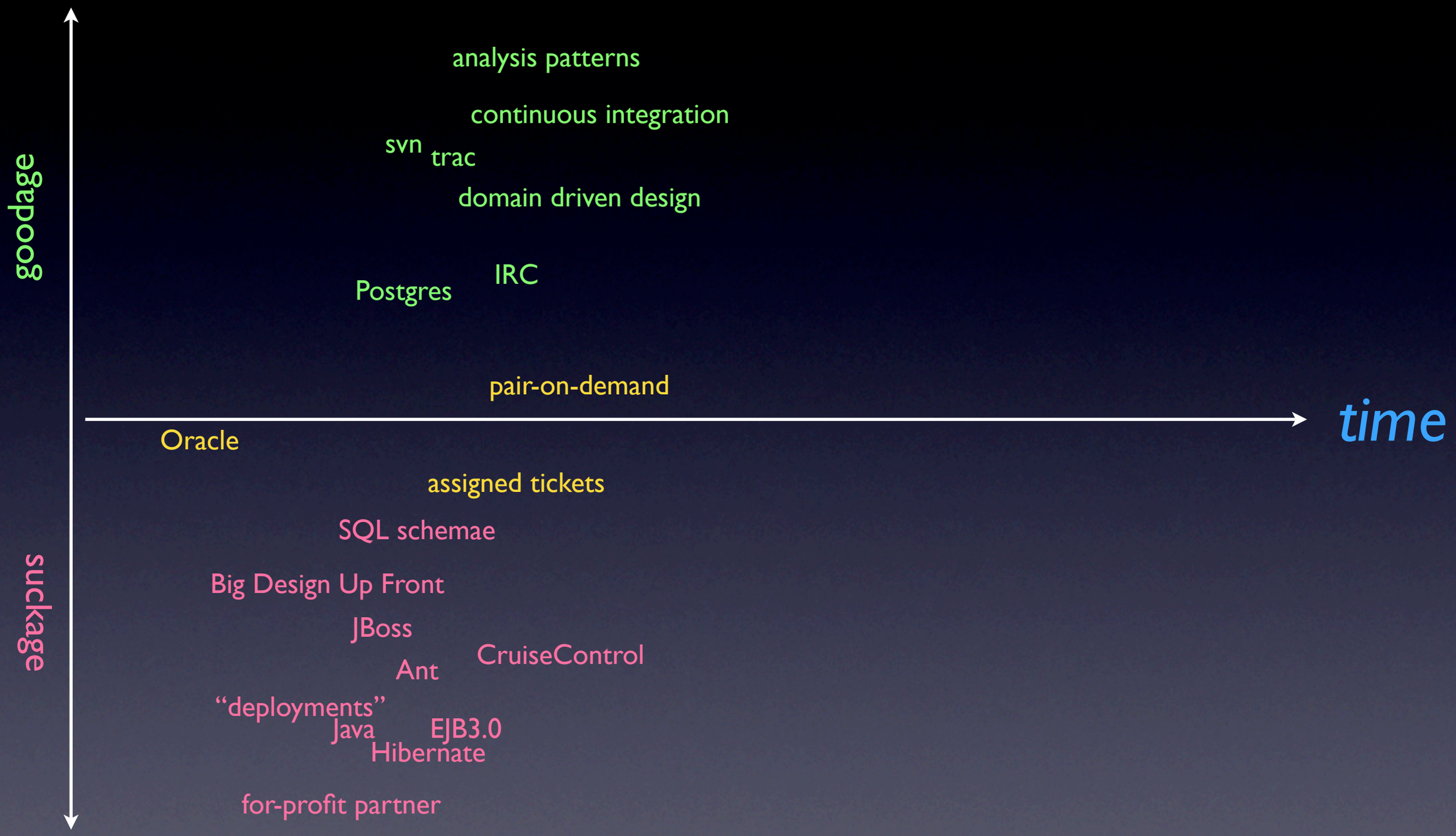
“deployments”

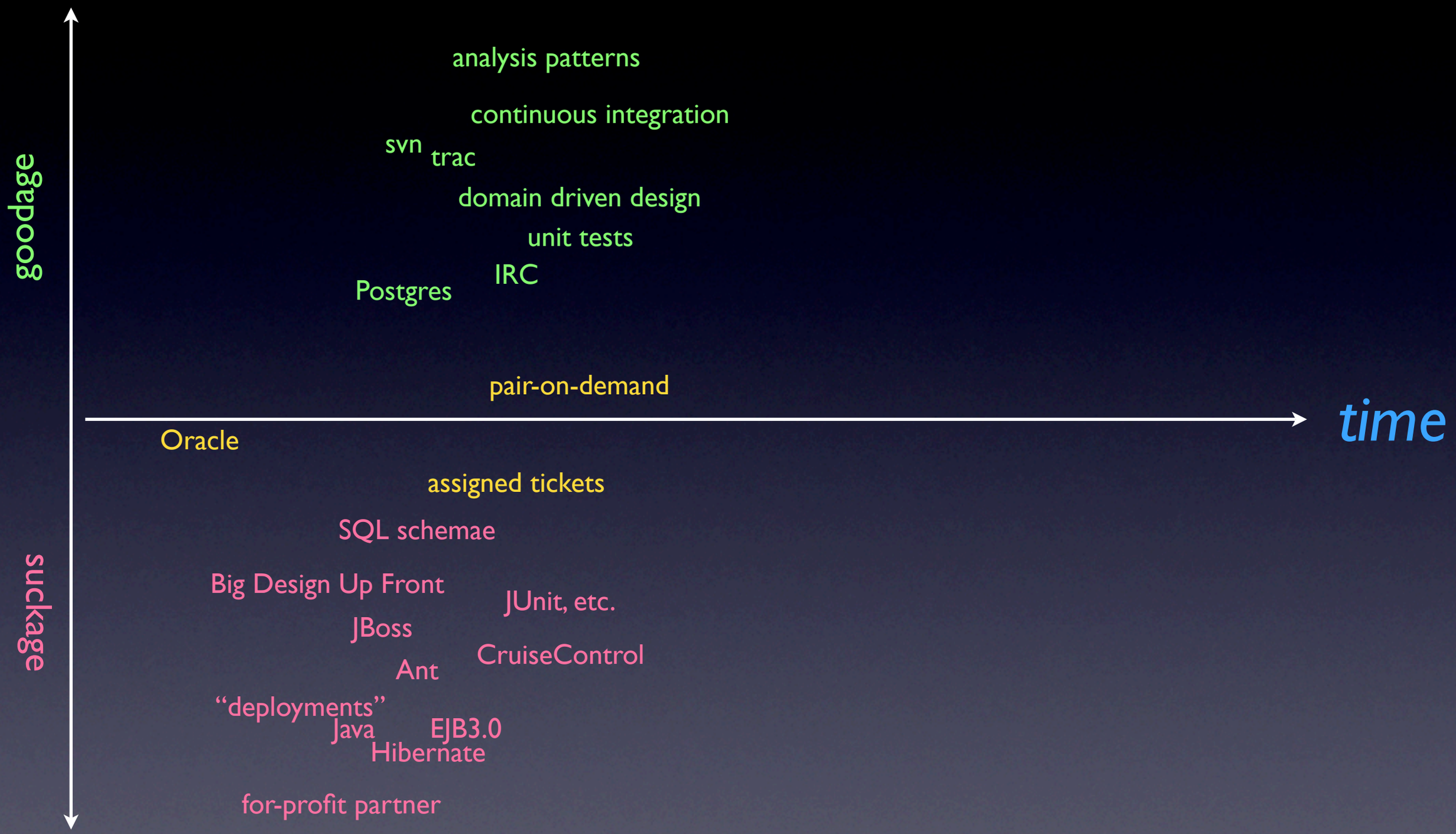
Java

EJB3.0

Hibernate

for-profit partner


goodage
suckage
time
analysis patterns
continuous integration
svn
trac
domain driven design
Postgres
IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
JBoss
CruiseControl
Ant
“deployments”
Java
EJB3.0
Hibernate
for-profit partner


goodage
suckage
time
analysis patterns
continuous integration
svn
trac
domain driven design
unit tests
IRC
Postgres
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
JUnit, etc.
JBoss
CruiseControl
Ant
“deployments”
Java
EJB3.0
Hibernate
for-profit partner

goodage

suckage

time

analysis patterns

continuous integration

svn

trac

domain driven design

unit tests

IRC

Postgres

auto deployments

pair-on-demand

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JUnit, etc.

JBoss

CruiseControl

Ant

Java

EJB3.0

Hibernate

for-profit partner

goodage

suckage

time

analysis patterns

continuous integration

svn

trac

public svn

domain driven design

unit tests

IRC

Postgres

auto deployments

pair-on-demand

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JUnit, etc.

JBoss

CruiseControl

Ant

Java

EJB3.0

Hibernate

for-profit partner

goodage

suckage

time

analysis patterns

continuous integration

svn

trac

public svn

domain driven design

unit tests

Postgres

IRC

AJAX

auto deployments

pair-on-demand

Oracle

assigned tickets

SQL schemae

Big Design Up Front

JUnit, etc.

JBoss

CruiseControl

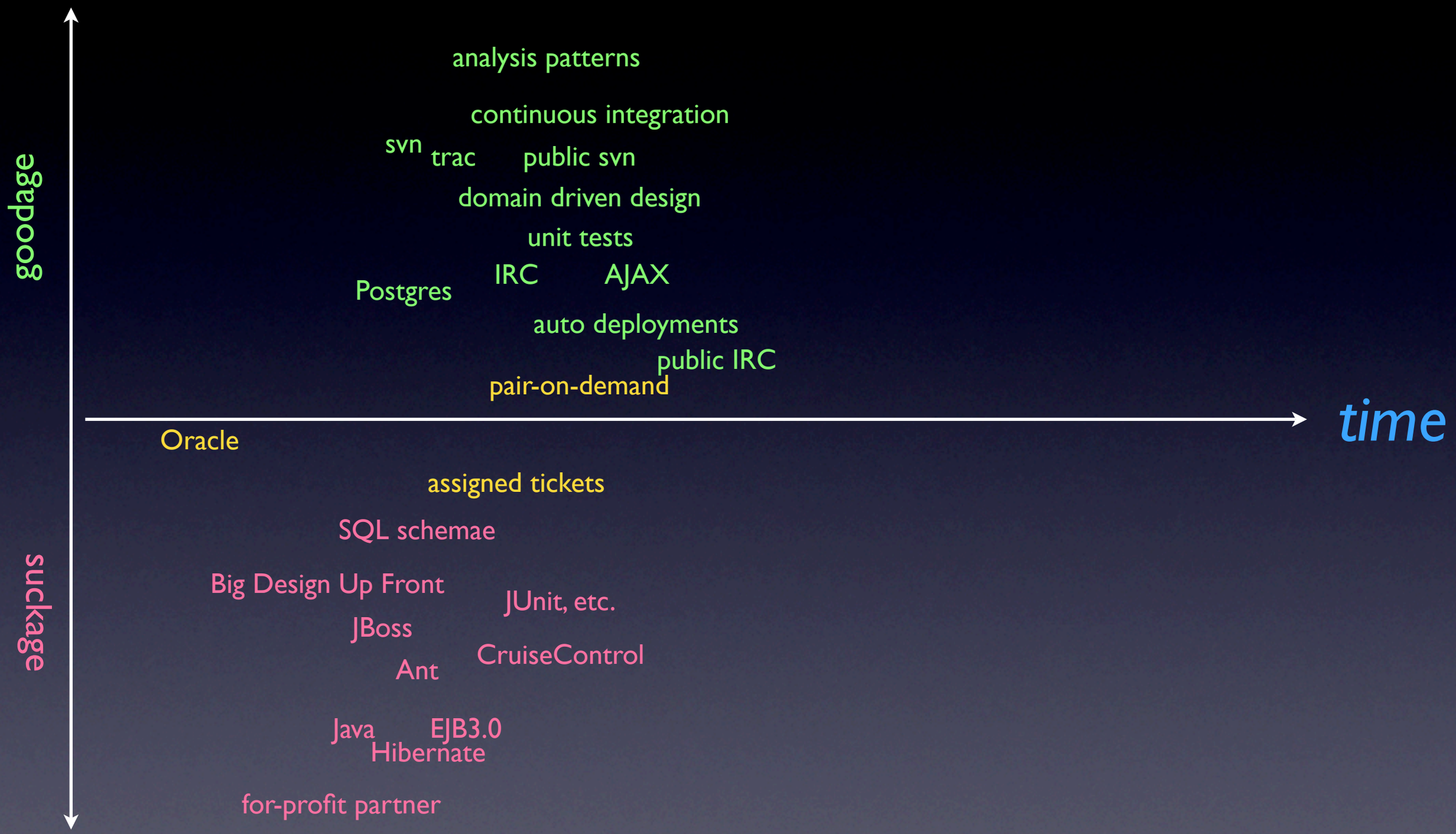
Ant

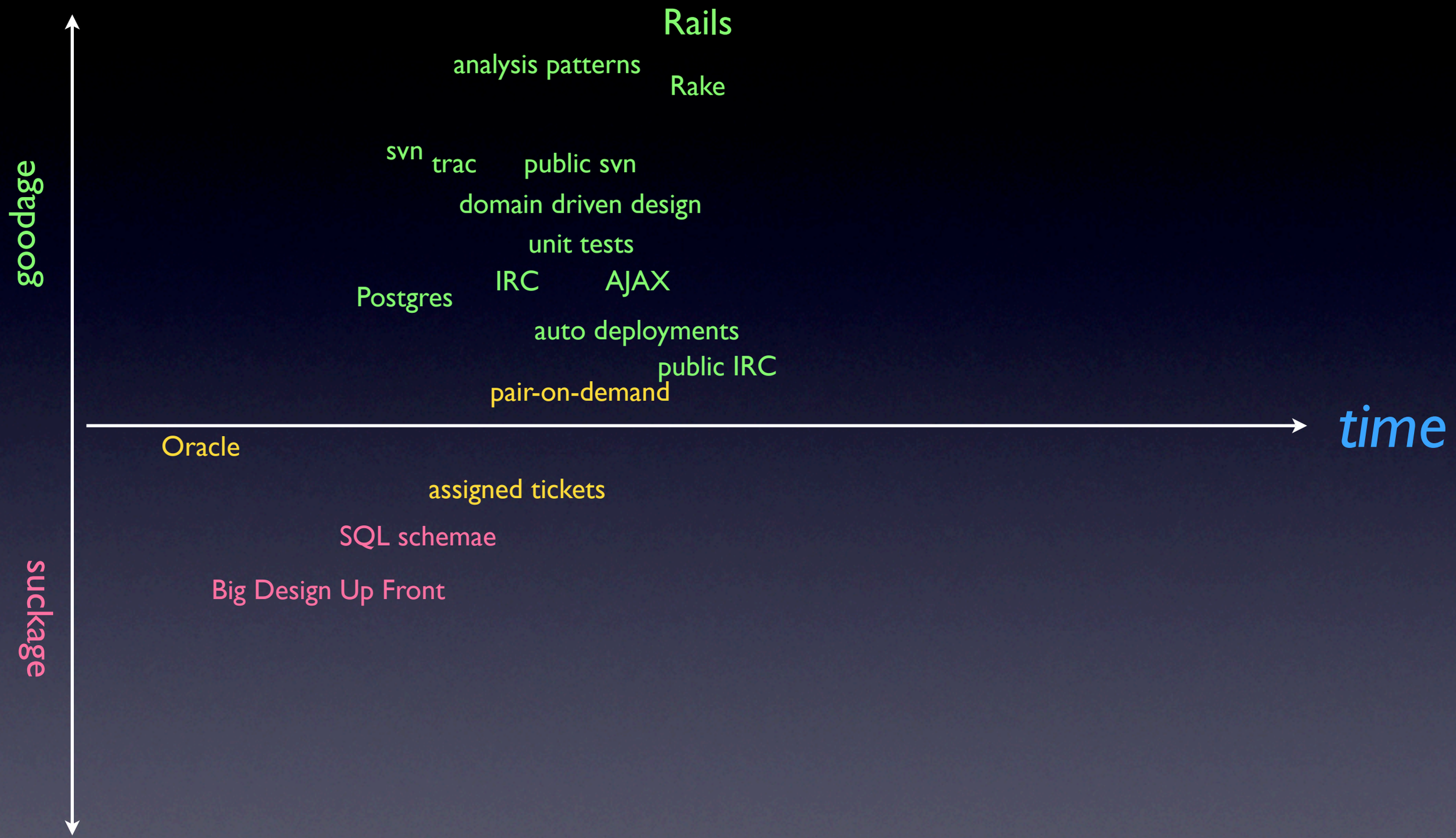
Java

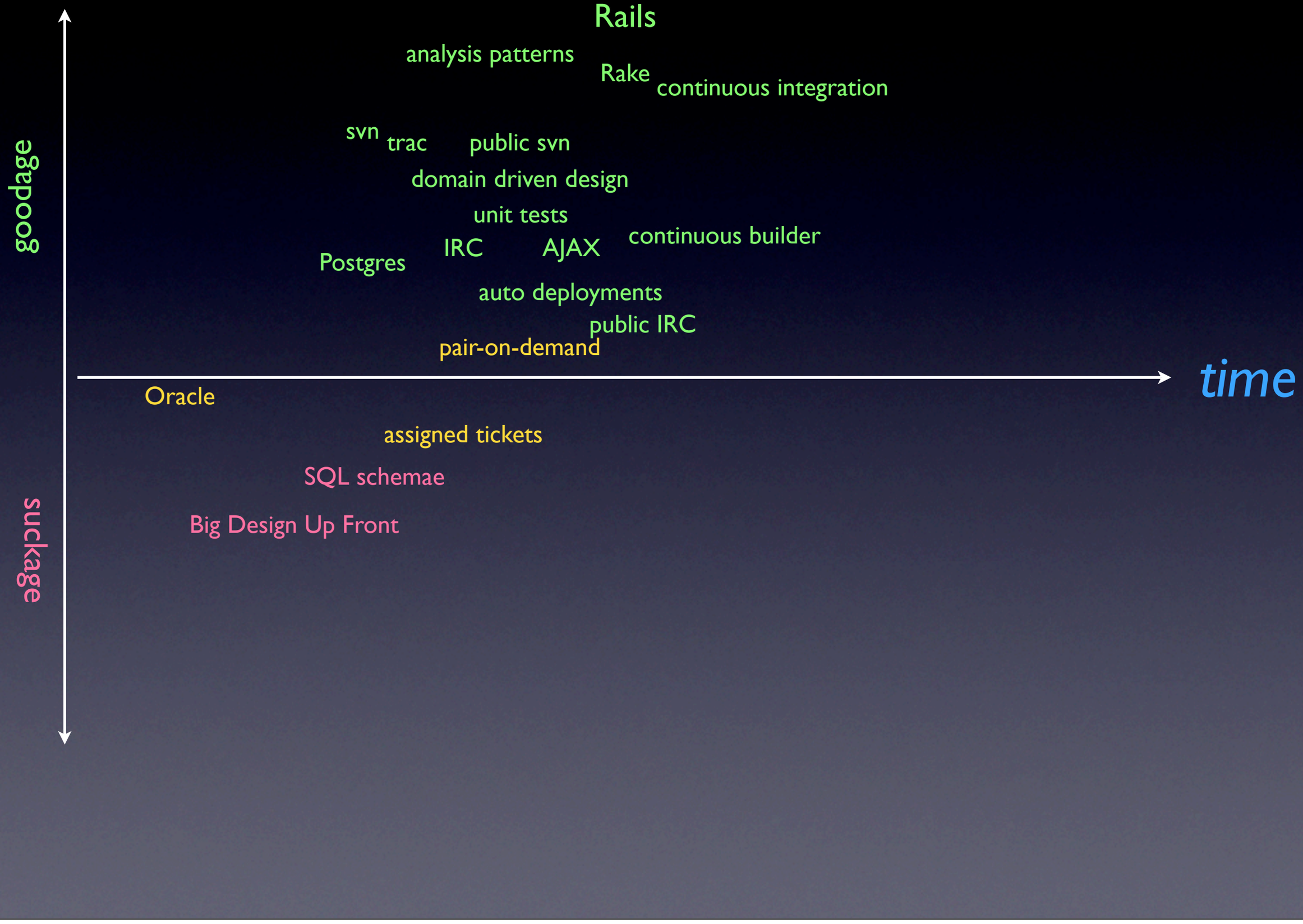
EJB3.0

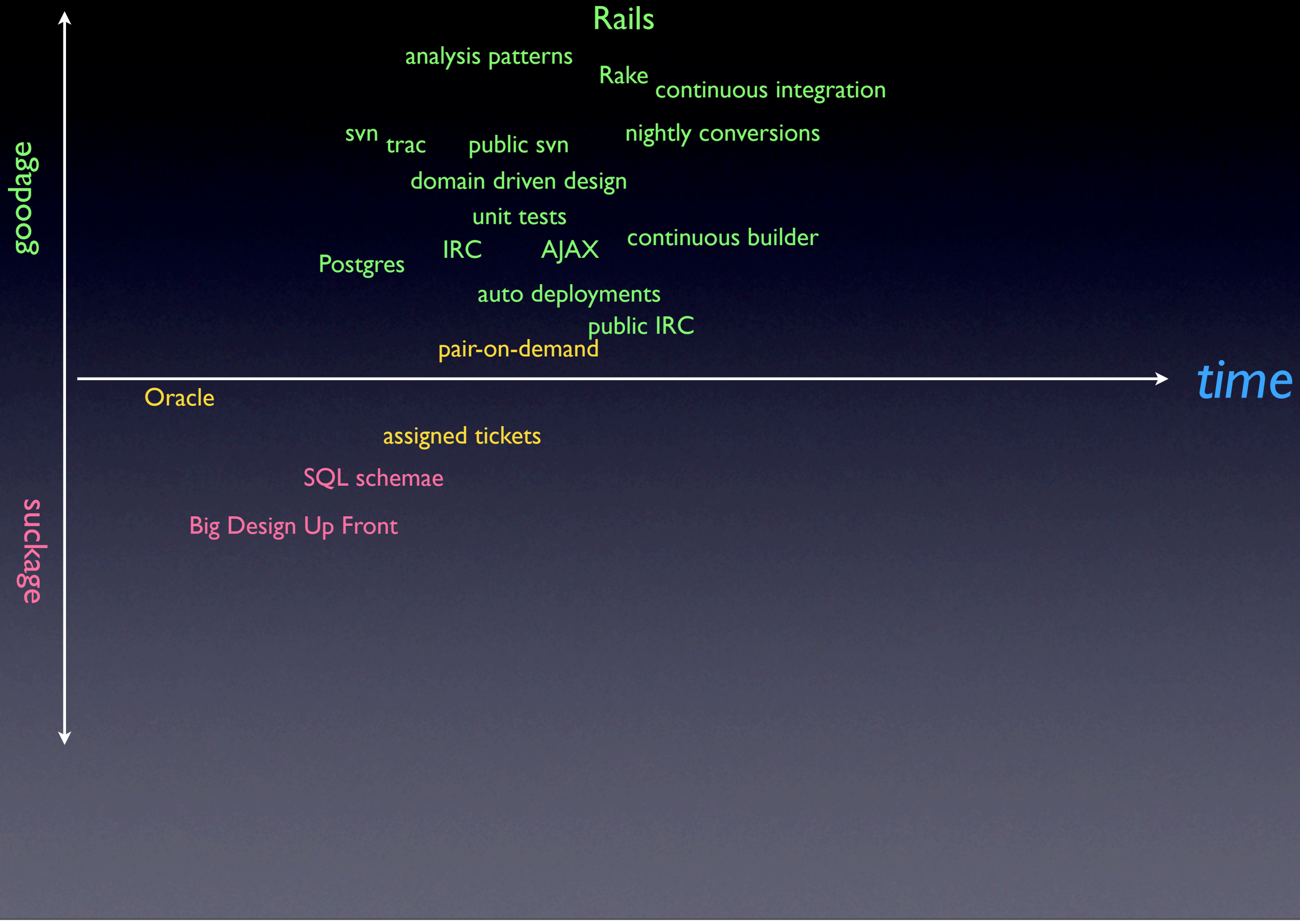
Hibernate

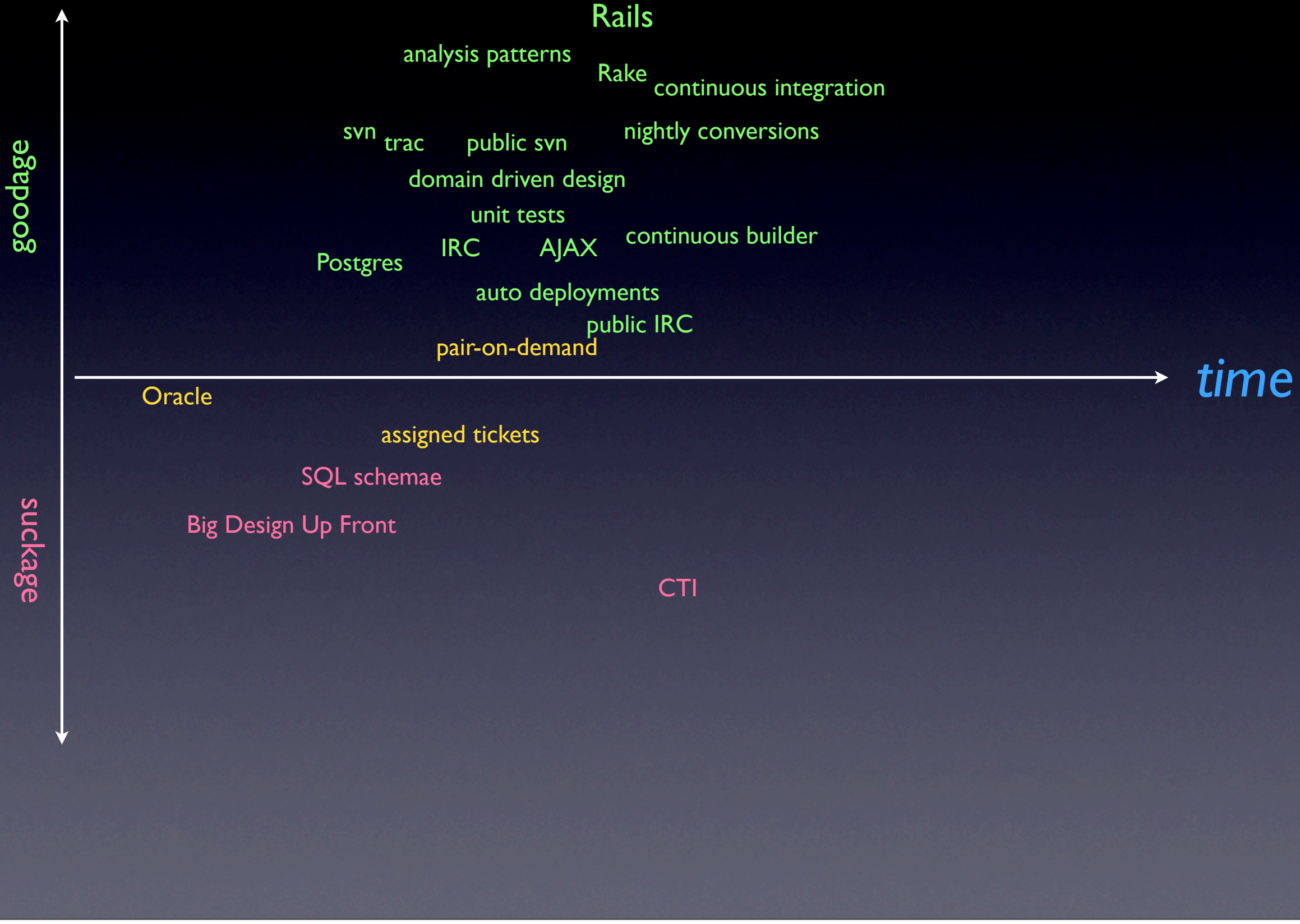
for-profit partner

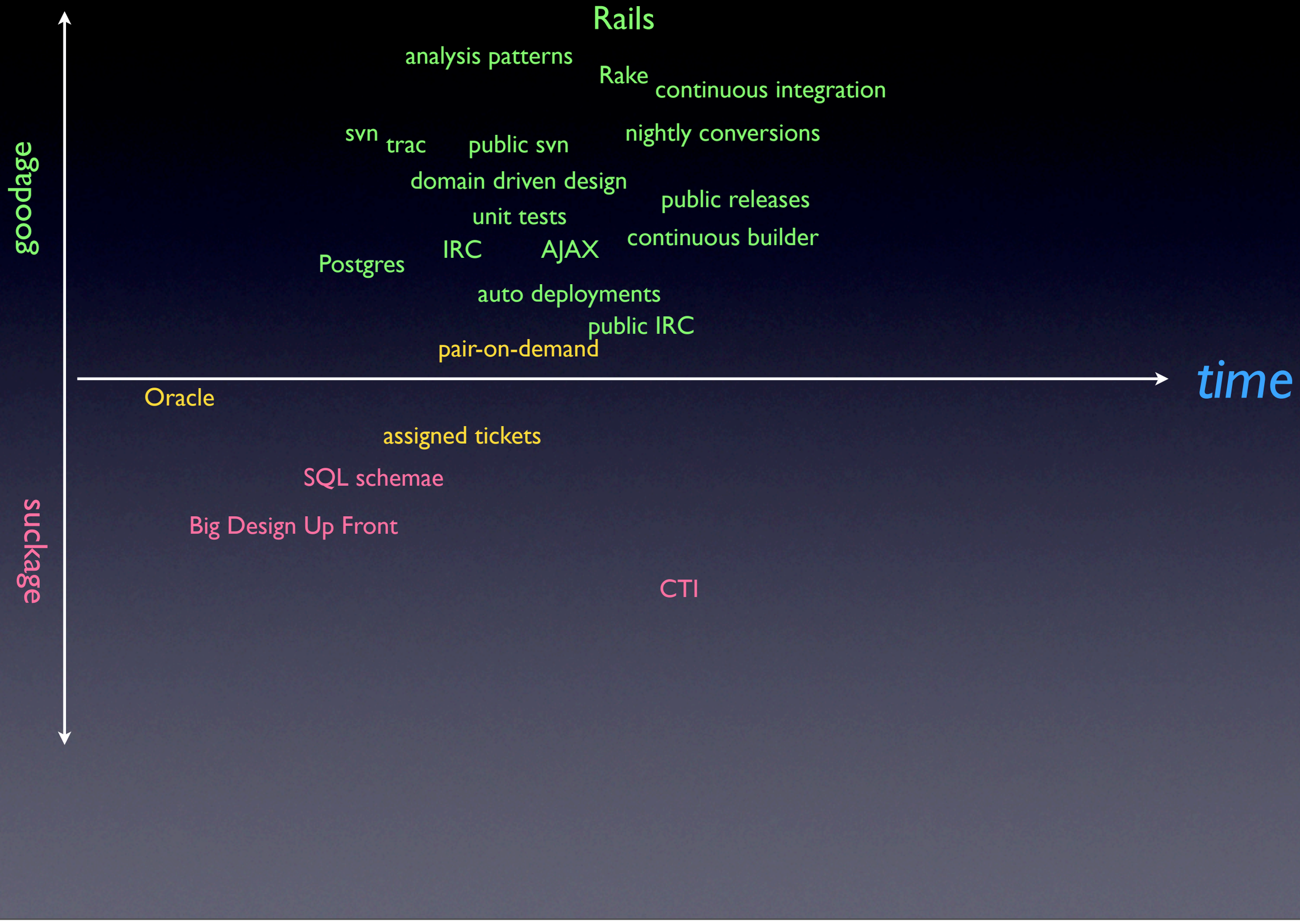

goodage
suckage
time
analysis patterns
continuous integration
svn
trac
public svn
domain driven design
unit tests
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
JUnit, etc.
JBoss
CruiseControl
Ant
Java
EJB3.0
Hibernate
for-profit partner

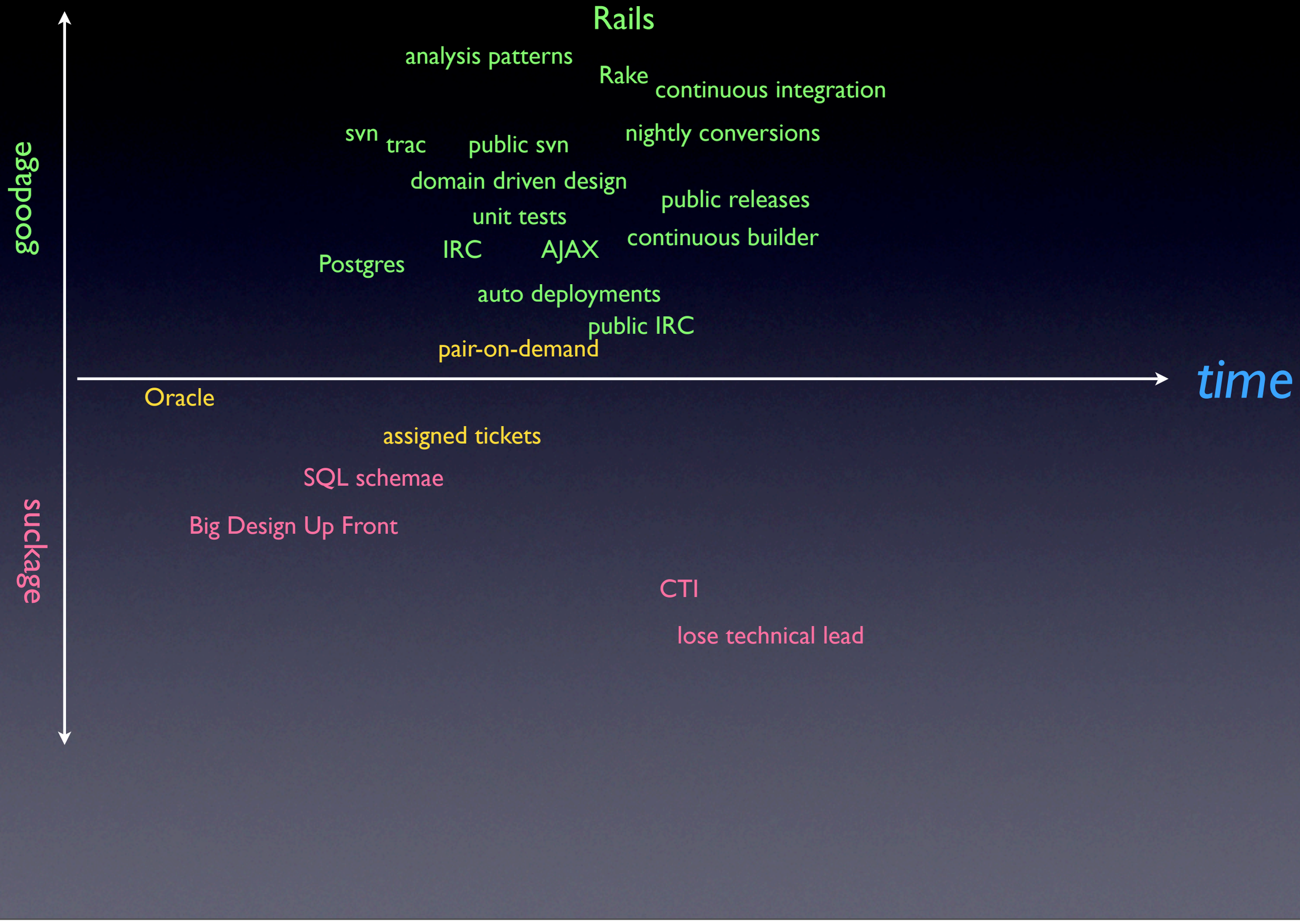

goodage
suckage
time
Rails
analysis patterns
Rake
svn
trac
public svn
domain driven design
unit tests
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front

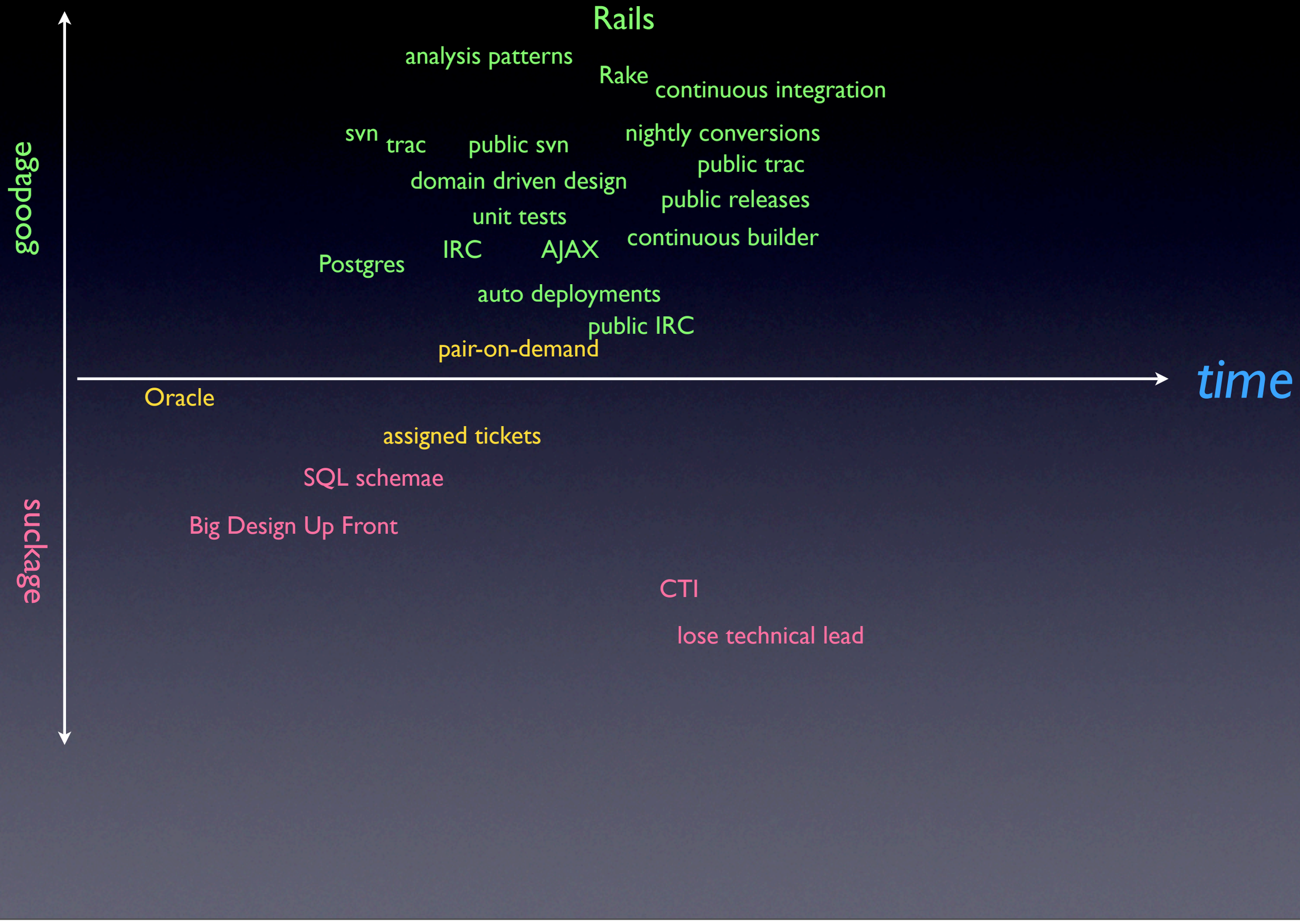

goodage
suckage
time
Rails
analysis patterns
Rake
continuous integration
svn
trac
public svn
domain driven design
unit tests
IRC
AJAX
continuous builder
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front

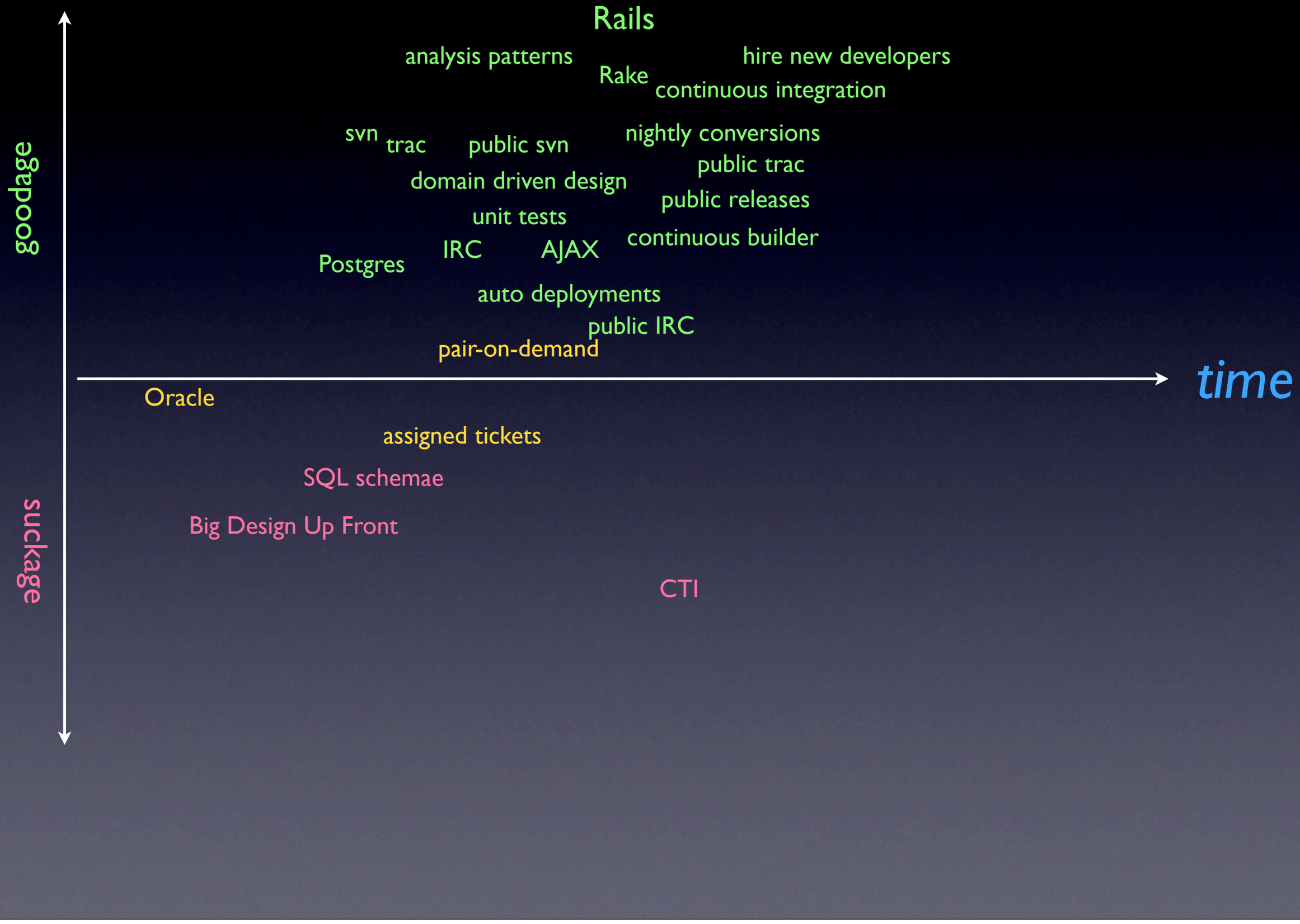

goodage
suckage
time
Rails
analysis patterns
Rake
continuous integration
svn
trac
public svn
nightly conversions
domain driven design
unit tests
continuous builder
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front

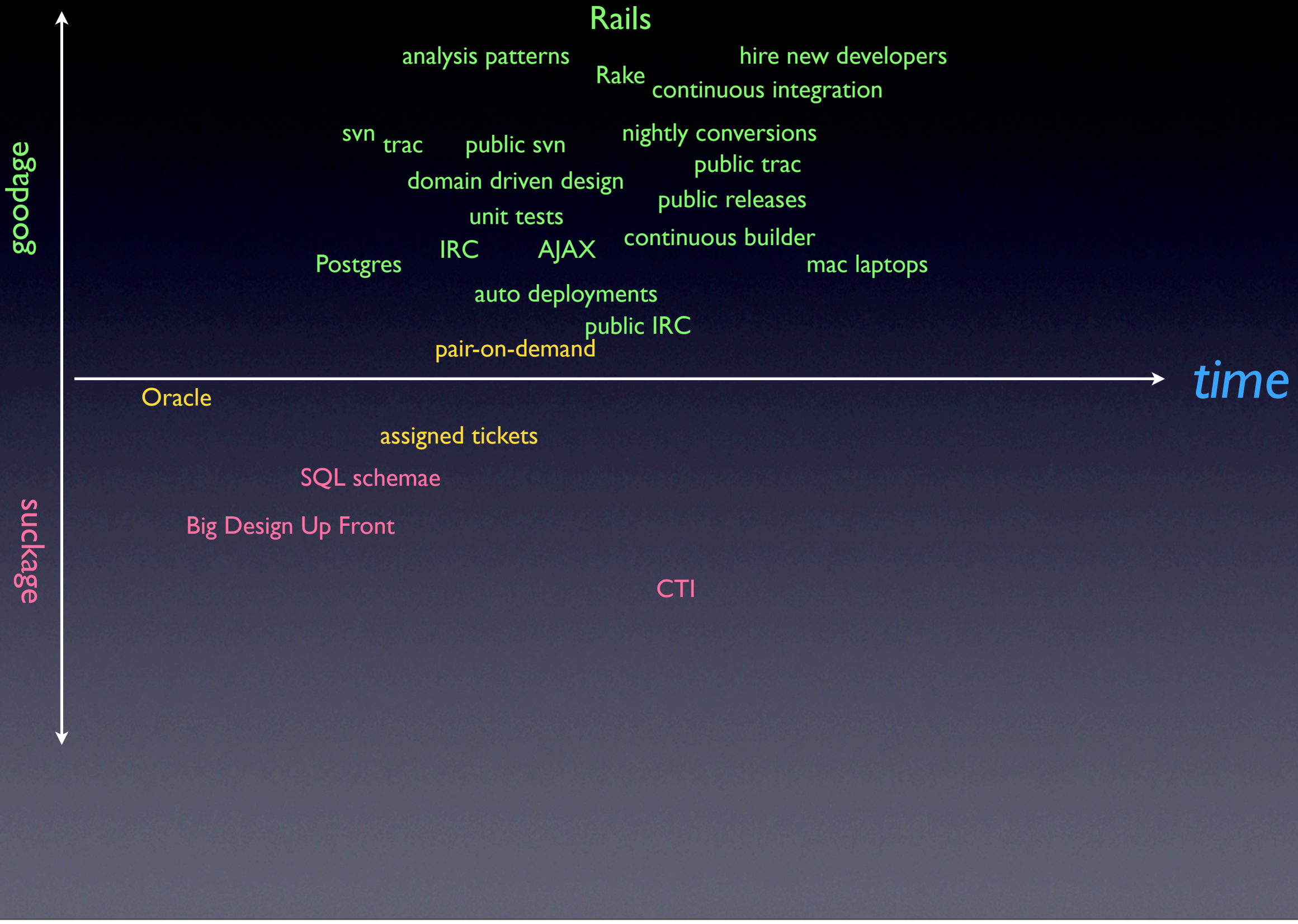

goodage
suckage
time
Rails
analysis patterns
Rake
continuous integration
svn
trac
public svn
nightly conversions
domain driven design
unit tests
IRC
AJAX
continuous builder
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
CTI

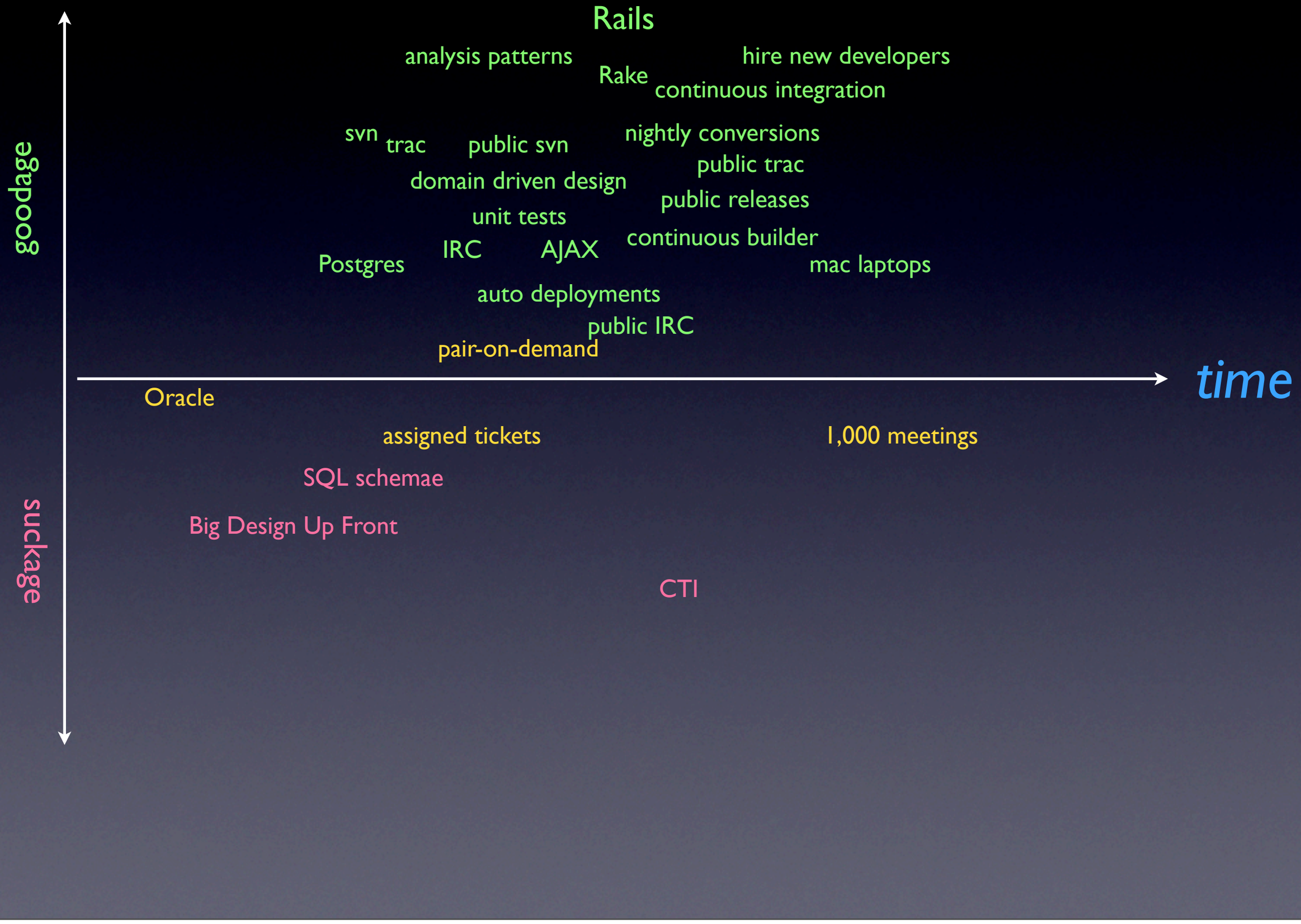

goodage
suckage
time
Rails
analysis patterns
Rake
continuous integration
svn
trac
public svn
nightly conversions
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
CTI

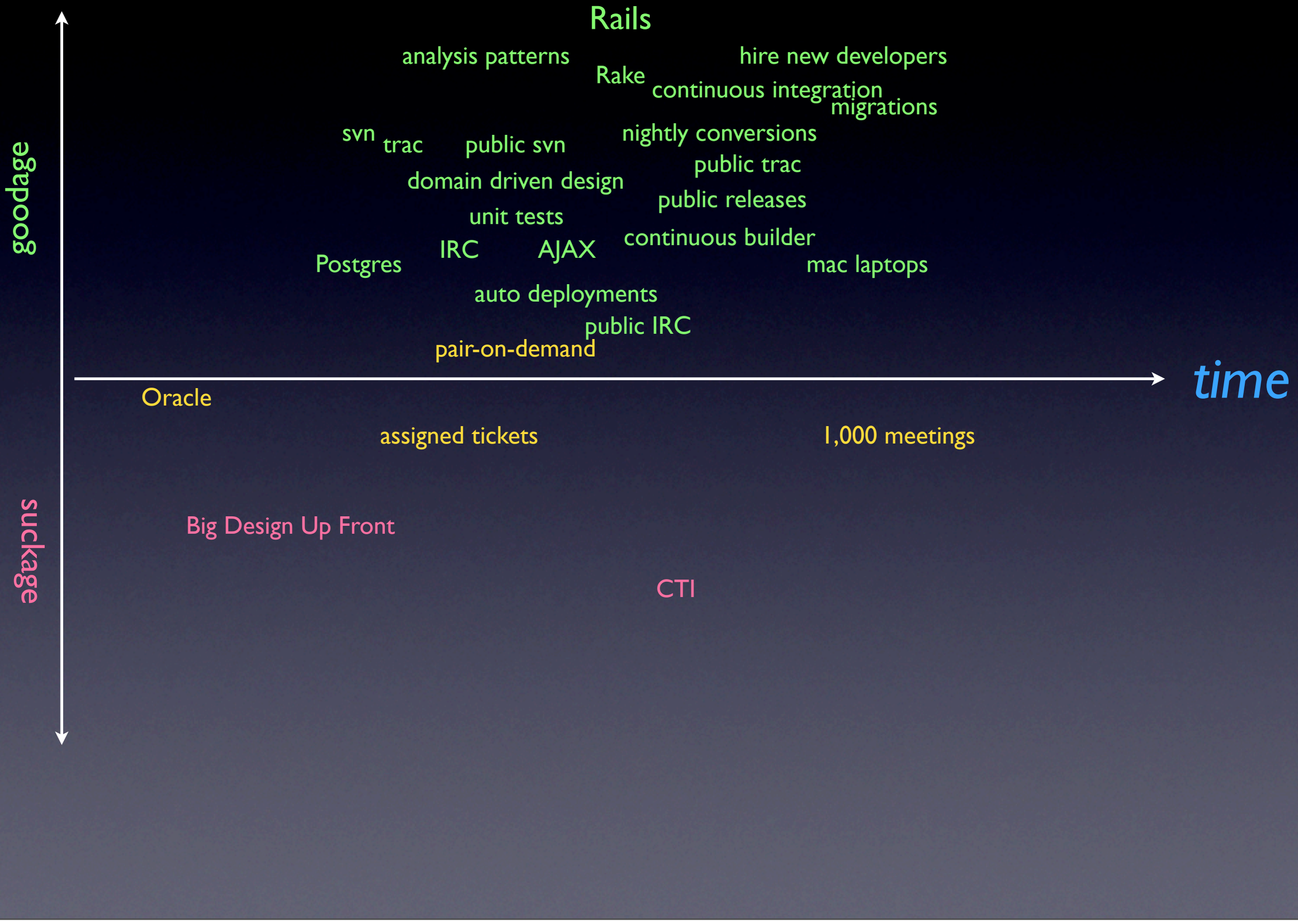

goodage
suckage
time
Rails
analysis patterns
Rake
continuous integration
svn
trac
public svn
nightly conversions
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
CTI
lose technical lead

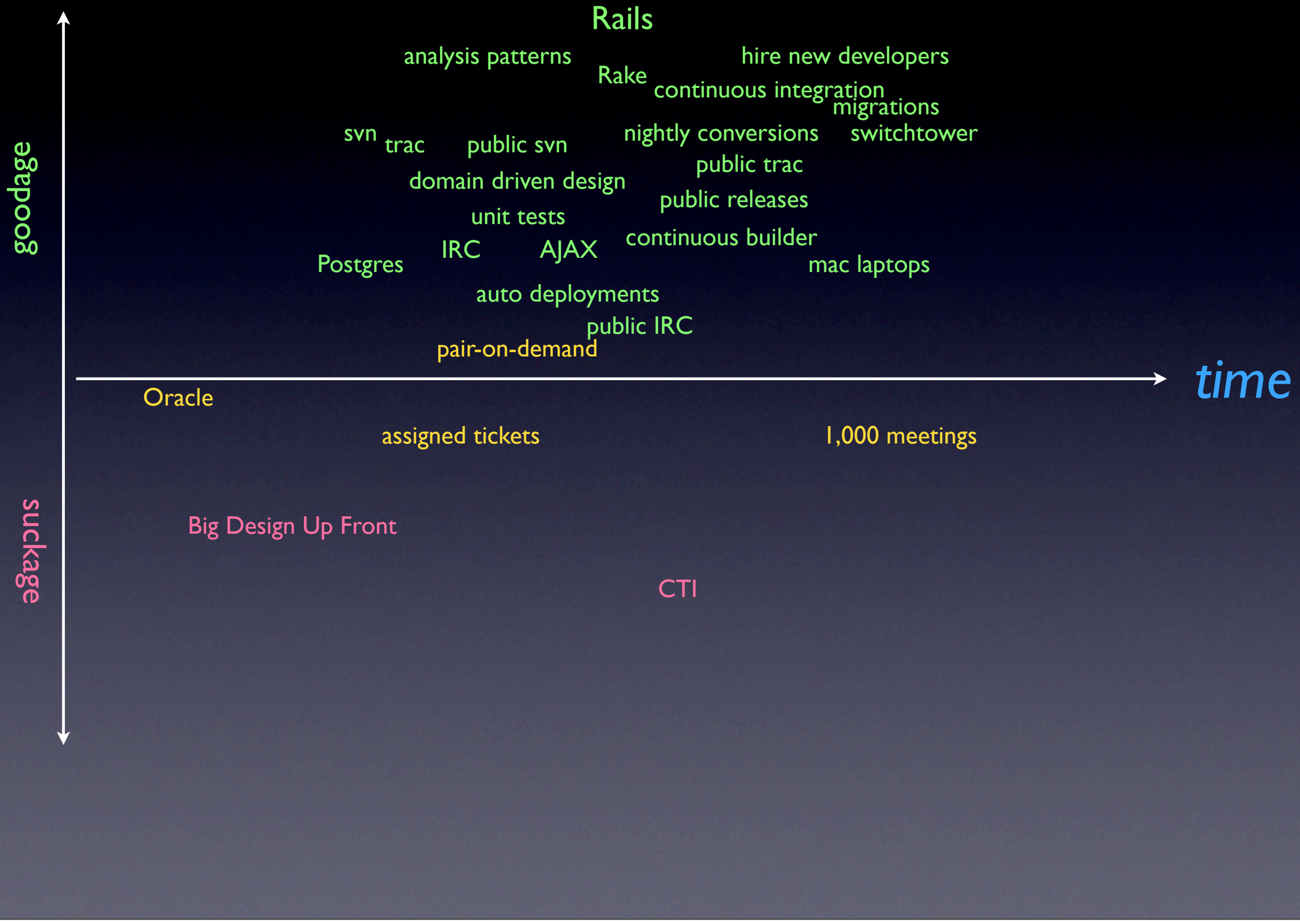

goodage
suckage
time
Rails
analysis patterns
Rake
continuous integration
svn
trac
public svn
nightly conversions
public trac
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
CTI
lose technical lead

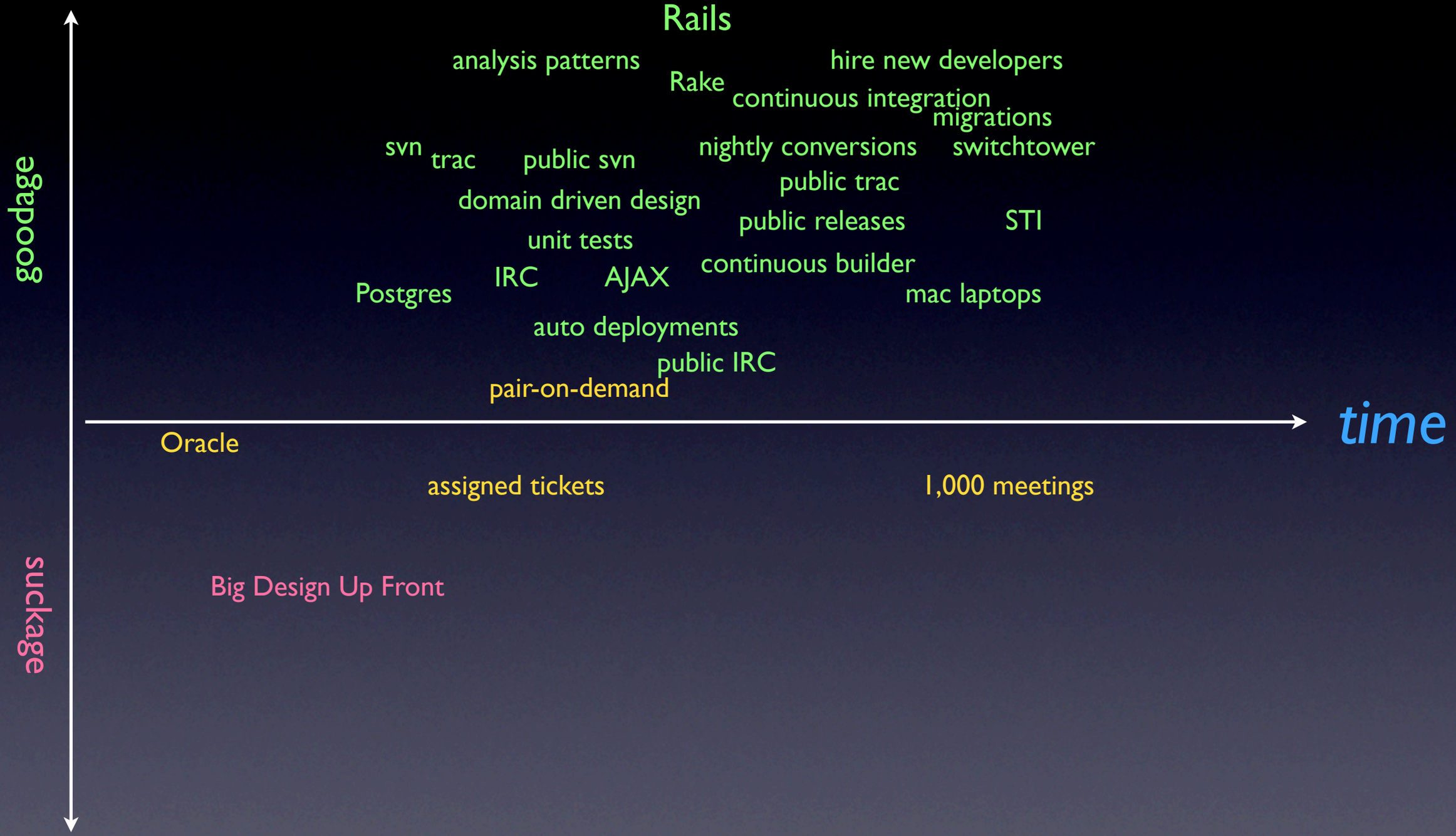

goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
svn
trac
public svn
nightly conversions
public trac
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
CTI

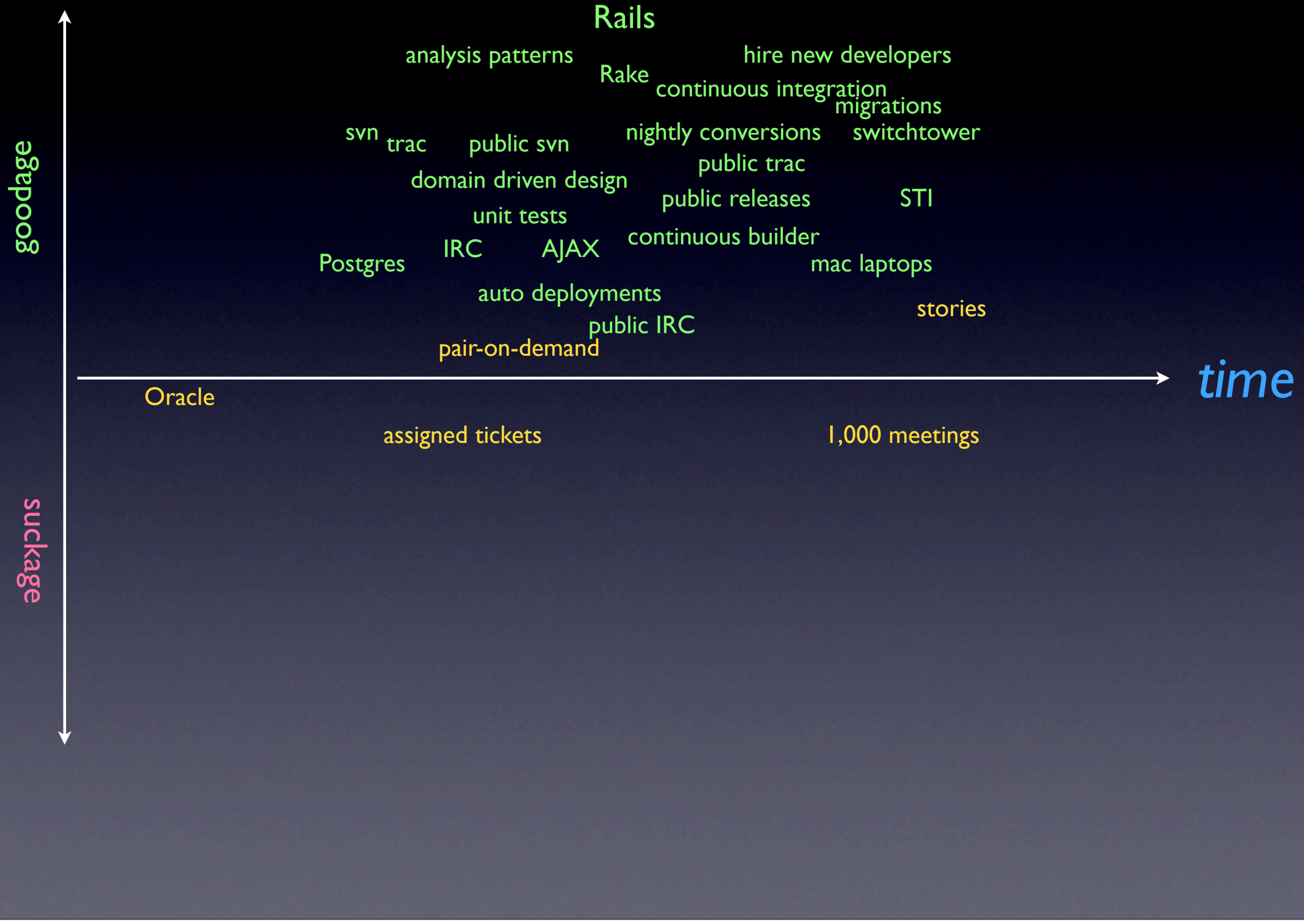

goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
svn
trac
public svn
nightly conversions
public trac
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
SQL schemae
Big Design Up Front
CTI

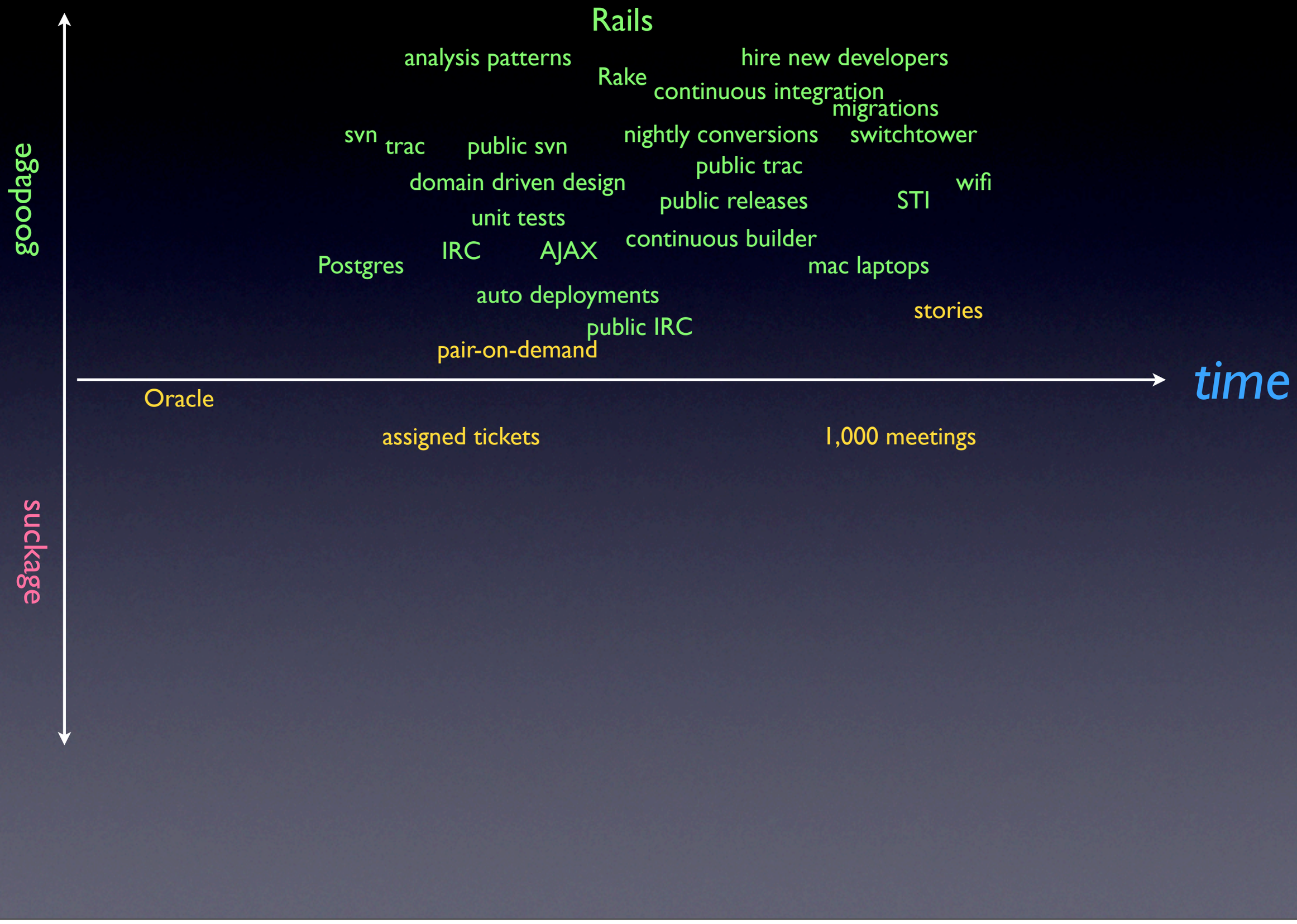

goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
svn
trac
public svn
nightly conversions
public trac
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings
SQL schemae
Big Design Up Front
CTI

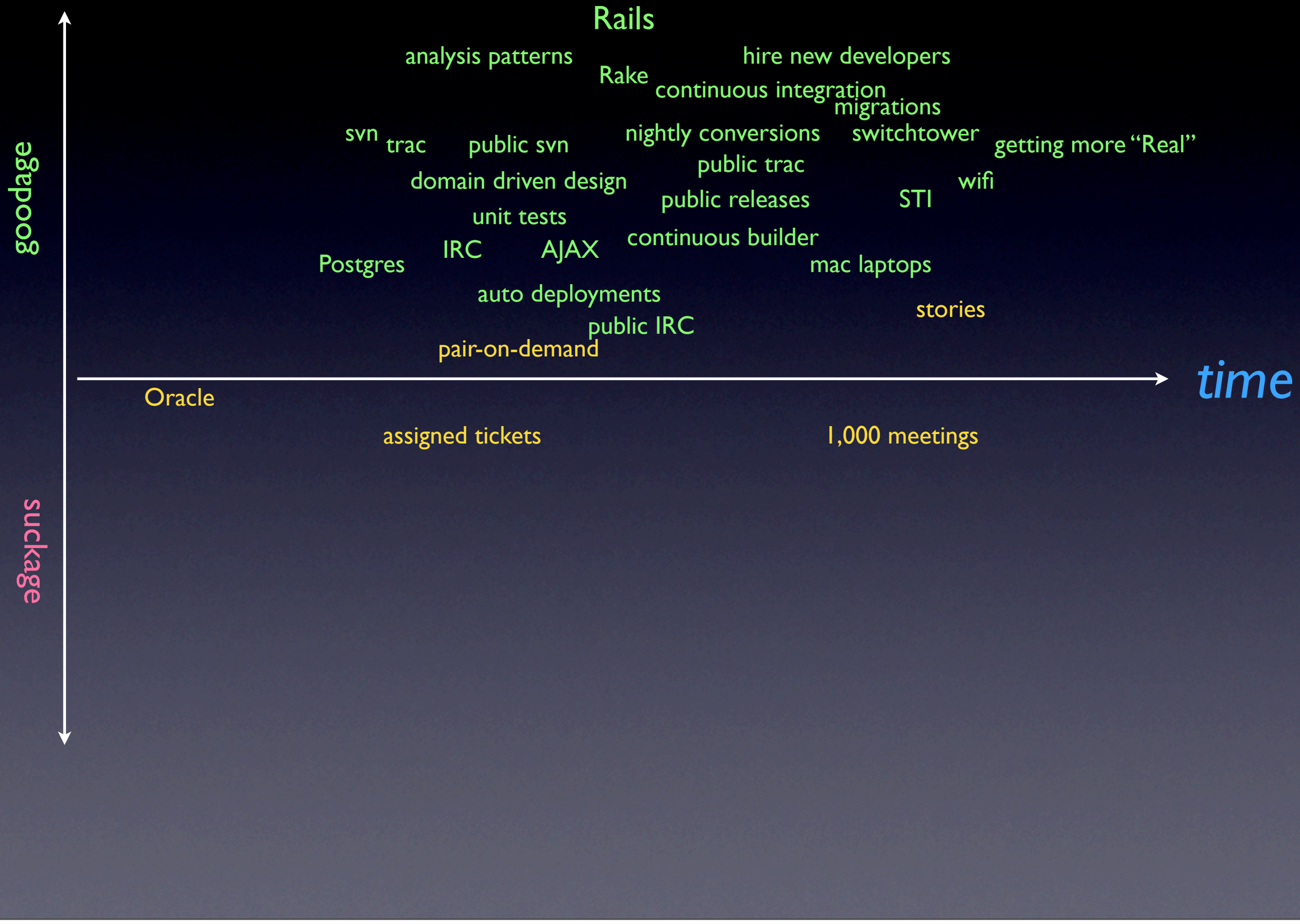

goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
public trac
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings
Big Design Up Front
CTI

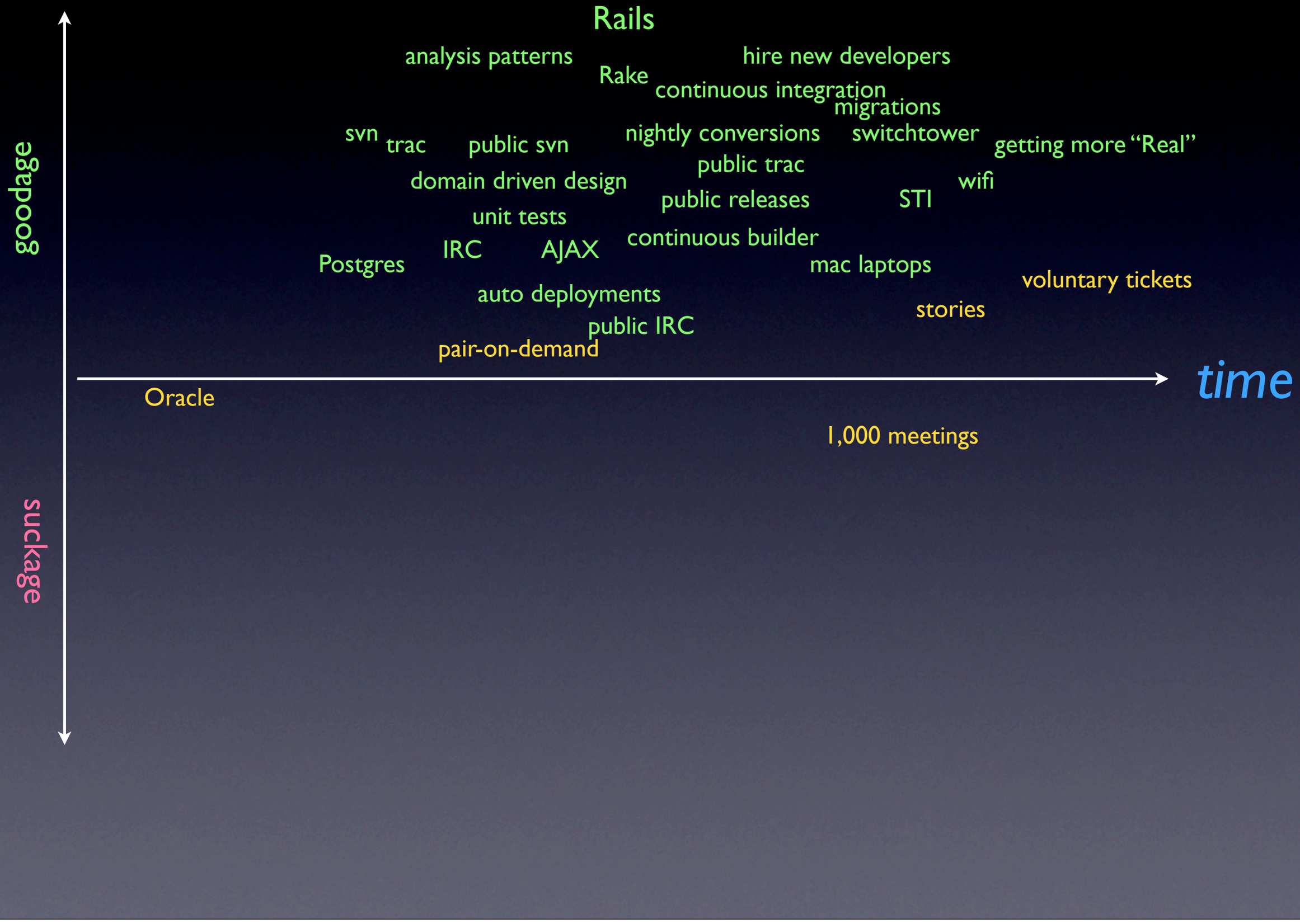

goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
public trac
domain driven design
public releases
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings
Big Design Up Front
CTI

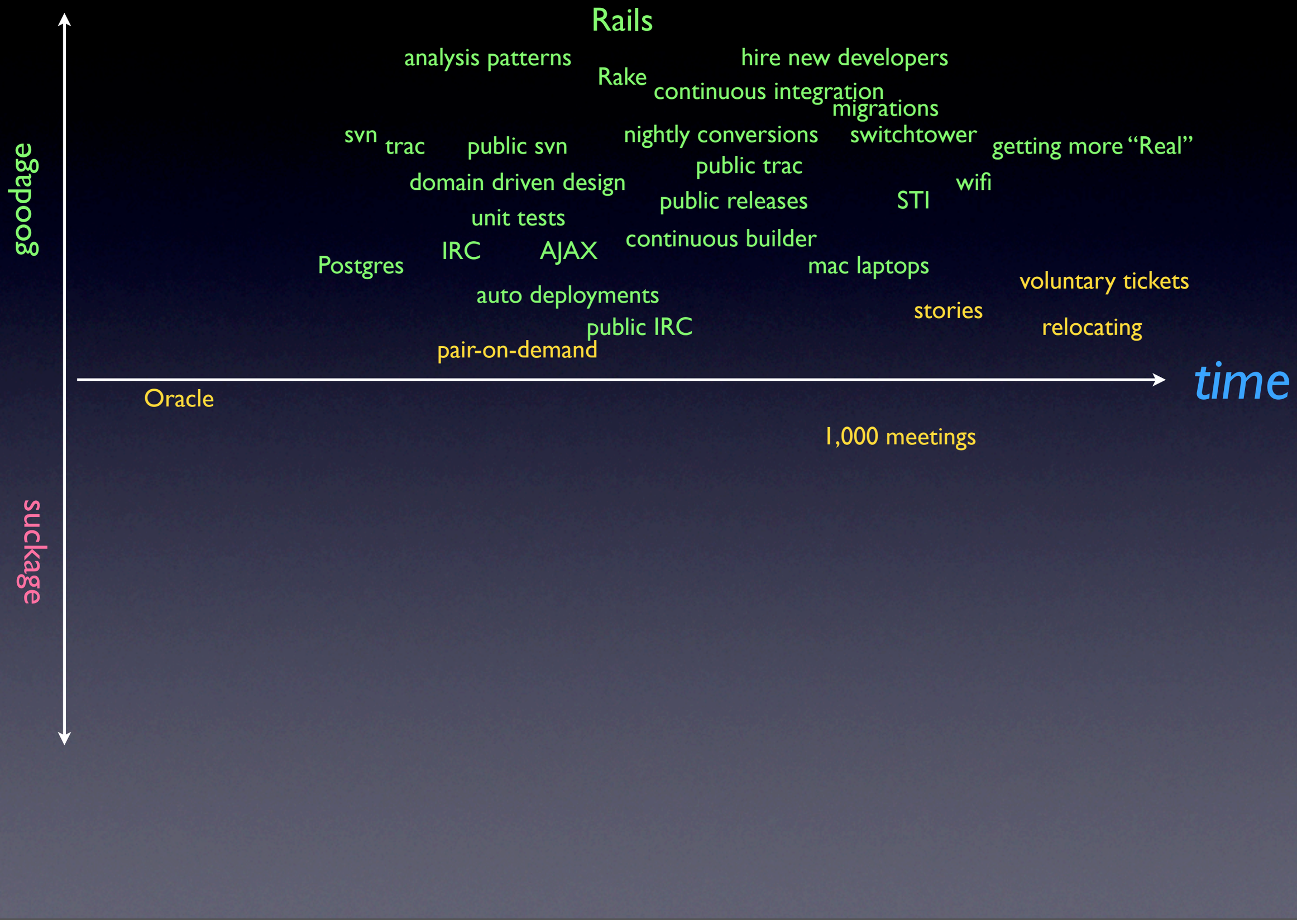

goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
public trac
domain driven design
public releases
STI
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings
Big Design Up Front


goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
public trac
domain driven design
public releases
STI
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
stories
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings


goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
public trac
domain driven design
wifi
public releases
STI
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
stories
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings


goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
getting more "Real"
public trac
domain driven design
wifi
public releases
STI
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
auto deployments
stories
public IRC
pair-on-demand
Oracle
assigned tickets
1,000 meetings


goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
getting more "Real"
public trac
domain driven design
wifi
public releases
STI
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
voluntary tickets
auto deployments
stories
public IRC
pair-on-demand
Oracle
1,000 meetings


goodage
suckage
time
Rails
analysis patterns
Rake
hire new developers
continuous integration
migrations
svn
trac
public svn
nightly conversions
switchtower
getting more "Real"
public trac
domain driven design
wifi
public releases
STI
unit tests
continuous builder
IRC
AJAX
Postgres
mac laptops
voluntary tickets
auto deployments
stories
public IRC
relocating
pair-on-demand
Oracle
1,000 meetings

Insights

The Rails Way

You really have to do things the Rails Way. All the little opinionated choices turn out to be right, whether they seem important or not. We fought pluralization for a long time. When we finally turned it on so many little headaches disappeared. When we were able to get onto Migrations (too early to start there) so many big headaches disappeared. We still keep foreign keys, auditing, various constraints, which are a big headache -- but at least we can have them in plugins and not have to think about them again.

Back in the fall I'd go to Healthcare Technology conferences and I'd mention to people that we were doing our system in Ruby on Rails and they'd say, 'Ruby on Rails... What's that?'

Back in the fall I'd go to Healthcare Technology conferences and I'd mention to people that we were doing our system in Ruby on Rails and they'd say, 'Ruby on Rails... What's that?'

Now when I go to conferences they say,
*"**Nobody** is using Ruby on Rails."*

Q&A